RANDOMIZED NUMERICAL LINEAR ALGEBRA FOR LARGE-SCALE MATRIX DATA

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Kun Dong

August 2019

RANDOMIZED NUMERICAL LINEAR ALGEBRA FOR LARGE-SCALE MATRIX DATA

Kun Dong, Ph.D.

Cornell University 2019

This dissertation is about computational tools based on randomized numerical linear algebra for handling large-scale matrix data. Since large datasets have become commonly available in a wide variety of modern applications, there has been an increasing demand for numerical methods for storing, processing, and learning from them. Matrices, the classical form for representing datasets, naturally connect these tasks with the rich literature of numerical linear algebra. For a diverse collection of problems, randomized methods offer extraordinary efficiency and flexibility. This work focuses on using randomized numerical linear algebra to build practical algorithms for problems of massive size and high complexity that traditional methods are unable to handle. Through this dissertation, we explore topics across network science, Gaussian process regression, natural language processing, and quantum chemistry. Our contribution includes a collection of scalable and robust numerical methods under a unifying theme, accompanied by efficient implementations. As a result, we are able to significantly speed up the computation for several existing applications, and explore problems and datasets that were intractable before.

BIOGRAPHICAL SKETCH

Kun Dong was born in Shaoxing, Zhejiang Province, China on September 26, 1992 to Jian Dong and Chamei Sang. Kun became interest in puzzles and mathematics in early elementary school, and participated in mathematics competitions until the end of high school.

In 2008, Kun moved to Newmarket, Ontario, Canada and attended Sir William Mulock Secondary School. He spent the next two years learning the new language and culture.

After graduation, Kun attended the University of California, Los Angeles in 2010 to study applied mathematics. During the summers of 2012 and 2013, he participated in the UCLA Applied Math REU program to work on dynamical system and crime modeling. He received tremendous help and encouragement on his way to graduate school from his REU mentors, Scott McCalla and James von Brecht. In 2014, Kun completed the departmental scholar program in mathematics, earning his B.S. and M.A. concurrently. He graduated Summa Cum Laude, receiving the departmental highest honor in applied mathematics and the Daus memorial award for his achievement in mathematics as an undergraduate.

In 2014, Kun was admitted to the Ph.D. program at Center for Applied Mathematics in Cornell University. He soon started working with Professor David Bindel, who later became his advisor. During the summer of 2017, he interned at the Lawrence Berkeley National Laboratory, where he worked under the mentorship of Professor Lin Lin. For the last three years at Cornell, he was partially supported by a National Science Foundation grant. In September 2019, Kun will move to Seattle and become a research scientist at Facebook, Inc.

This document is dedicated to my parents and wife.

ACKNOWLEDGEMENTS

My advisor David Bindel has my deepest appreciation, for this work is only possible with his guidance. I thank him not only for sharing his vast knowledge and experience, but also for the patience, gentleness, and encouragement in his mentorship. I will forever carry the lessons I learned from him.

I want to thank my committee members Andrew Wilson and Adrian Lewis, as well as my temporary advisor Shane Henderson. Andrew led me into an important area of my research, which makes up Chapter 3 of this dissertation. Adrian and Shane both offered my valuable advice in research and life as a graduate student.

I deeply appreciate all my wonderful collaborators. Chapter 2 was co-authored with Austin Benson and David Bindel. Chapter 3 was co-authored with David Bindel, David Eriksson, Eric Lee, Hannes Nickisch and Andrew Wilson. Chapter 4 was co-authored with David Bindel, Sungjun Cho, Moontae Lee, and David Mimno. Chapter 5 was co-authored with Wei Hu and Lin Lin. Thank you for the time and effort in working with me. It was truly a pleasure to collaborate with you all.

I also want to thank all the students and staffs at CAM, especially Andrew Loeb, Zachary Clawson, Eric Lee, David Eriksson, and my best friend Marc Aurèle Tiberius Gilles. You made this part of my life not just rewarding, but enjoyable as well.

I would like to acknowledge my colleagues during my internships at Lawrence Berkeley National Laboratory and Google Network Infrastructure Team, particularly my mentors Lin Lin and Rui Wang. You taught me many useful skills, and helped me become a better researcher and programmer.

Most importantly, I am sincerely grateful for my supportive family and wife. You have always been by my side even when I am far away.

My work has been supported by the Cornell Center for Applied Mathematics, the National Science Foundation Grant DMS-1620038, teaching assistantship in Cornell's

Department of Computer Science and Department of Mathematics, and the Cornell University Fellowship. During the summer of 2017, I was supported by the Lawrence Berkeley National Laboratory as a graduate research intern.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Today massive datasets appear in countless applications across scientific fields. Empowered by modern computational infrastructure, researchers have spent tremendous effort to take advantage of the abundance of information. Machine learning, as one of the fields that rely heavily on large data, has witnessed incredible development and had far-reaching impact on nearly every area of scientific discovery. Even in traditional fields, such as partial differential equations, researchers are working on larger and more complex systems, resulting in bigger datasets. Typically, matrices arise as the natural representation of data in both explicit and implicit ways. Object-feature matrices are the most common type; second-order models, such as co-occurrence matrices and kernel matrices, capture the relationship between pairs of objects; Laplacian matrices and other linear operators describe dynamics on objects. In most instances, the matrices grow in size with the datasets; thus, manipulating them becomes an inevitable challenge. Numerical linear algebra (NLA), the study of matrix-based numerical methods, promptly becomes the driving force behind many of these applications.

Similarly, this ongoing trend of big data has led NLA into new and exciting directions. While deterministic, factorization-based methods usually have strong theoretical guarantees and highly optimized implementations, emerging problems have far exceeded the scale they are designed for, even with present-day computational power. As a result, a great number of novel algorithms centered around scalability have been developed in response to this increasing demand. Randomized methods provide one particular approach that brings efficiency through introducing stochasticity. Randomized NLA aims at building fast algorithms that return sufficiently good approximate solutions. Both "fast" and "good" are measured relatively depending on the underlying

problem, but the results are usually supported by matrix perturbation theory and probability theory. Even though scalability is the foremost objective, randomized NLA also designs algorithms around other criteria, e.g., interpretability, and robustness. Drineas and Mahoney [54] provide a great overview of recent developments in randomized NLA.

This dissertation applies randomized NLA to large-scale data matrix in two settings:

1. The matrix is data sparse, i.e., it can be described with far fewer than $O(NM)$ parameters, where $N \times M$ is its size. The goal is to efficiently obtain a compressed representation of the matrix, ideally in an interpretable way.
2. Only partial information from the matrix is required. The matrix may be explicitly available, but the extraction of information is expensive; or the matrix is implicit, due to the cost of formation or storage.

These two scenarios are not mutually exclusive. For example, Chapter 4 covers the computation of the log-determinant for kernel matrices, where we can often take advantage of their low-rank plus diagonal structure. The applications we work with come from network science, Gaussian process regression, natural language processing, and quantum chemistry. The diversity in background alone is a compelling evidence for the versatility of randomized NLA. The outline is detailed below.

Chapter 2 covers some mathematical preliminaries as well as common topics shared between the following chapters.

Chapter 3 studies the spectral densities of massive real-world graphs. We borrow tools developed in condensed matter physics, and add novel adaptations to handle the spectral signatures of common graph motifs. The resulting methods are highly efficient, as we illustrate by computing spectral densities for graphs with over a billion edges on a

single compute node. Beyond providing visually compelling fingerprints of graphs, we show how the estimation of spectral densities facilitates the computation of many common centrality measures, and use spectral densities to estimate meaningful information about graph structure that cannot be inferred from the extremal eigenpairs alone. This work is accepted for publication in KDD 2019 [53].

Chapter 4 develops novel approaches for Gaussian process (GP) regression. The computational bottleneck of kernel learning for GPs is computing the log determinant and its derivatives of an $N \times N$ kernel matrix. Building on existing fast matrix-vector-multiplication approximation for kernel matrices, we combine iterative methods with stochastic estimation to lower the cost from $O(N^3)$ to $O(N)$. The resulting methods are highly efficient and flexible, allowing us to work with datasets much larger than the traditional GP capability. Furthermore, we extend these ideas to GP regression on both function values and derivatives. Our approaches, together with dimensionality reduction and preconditioning, let us scale Bayesian optimization with derivatives to high-dimensional problems and large evaluation budgets. This work appeared in NeurIPS 2017 [51] and NeurIPS 2018 [59].

Chapter 5 proposes a complete pipeline for spectral inference of topic models that scales well with both the size of the vocabulary and the dimension of the latent space. It allows us to simultaneously compress and rectify the co-occurrence statistics, then learn latent variables directly from the compressed form with little loss of precision. We verify that our methods are as accurate as previous approaches on both textual and non-textual data, and run much faster. The work is in submission.

Chapter 6 describes how to use centroid Voronoi tessellation to accelerate electronic structure calculation. The recently-developed interpolative separable density fitting decomposition compresses the redundant information in electron orbital pairs through a set

of non-uniform interpolation points. Our method, implemented as a weighted K-means algorithm with random initialization, achieves comparable accuracy to the existing procedure but at a cost negligible in the overall calculations. We also find that our algorithm, as a continuation method, enhances the smoothness of the potential energy surface. This work appeared in Journal of Chemical Theory and Computation [52].

CHAPTER 2

PRELIMINARIES

2.1 Data as Matrices

Matrices are ubiquitous in data-related fields, such as computer science, statistics, and machine learning. From different perspectives, a matrix can represent a variety of objects:

- A simple and compact way of storing information. For example, each column of a matrix corresponds to a data point and each row stands for a feature.
- A linear map $A : \mathcal{V} \to \mathcal{U}$ from one vector space to another through matrix-vector-multiplication $v \mapsto u = Av$.
- A bilinear map $A : \mathcal{V} \times \mathcal{U} \to \mathbb{R}$ such that $(v, u) \mapsto u^T Av$.

Most importantly, these representations are not incompatible with each other, and it is the capacity for multiple simultaneous interpretations that makes matrices such a powerful tool in applications. For instance, an adjacency matrix $A \in \mathbb{R}^{N \times N}$ can describe an unweighted graph where $A_{ij} = 1$ if node i and j are connected by an edge, and 0 otherwise. Each row of A can be viewed as the connectivity for the corresponding node. Alternatively, applying A to a vector standing for the initial condition of a path counting process captures the dynamics on this graph. Finally, the bilinear map of A on $\mathcal{V} = \mathcal{U} = \{0, 1\}^N$ measures the number of edges from one set of nodes to the other and thus can be used in graph partitioning tasks.

Therefore, it is inadequate to categorize data matrices by their linear algebraic roles. Instead, we introduce the matrices encountered in this dissertation by the properties of

datasets they encapsulate. Regardless of the type of data matrices we are dealing with, the most critical task is identifying the intrinsic numerical properties attached to it, so that we can adapt our algorithms to specific applications.

Matrices for Features An object-feature matrix $A \in \mathbb{R}^{M \times N}$ encodes N data points as column vectors, each with M features. Many algorithms involve linear algebraic operations on data, and such a representation naturally facilitates their computation. For example, data compression and dimensionality reduction techniques, such as frequency sampling and principal component analysis, are usually implemented as low-rank decompositions on data matrices. Neural networks can be viewed as a sequence of linear transformations and element -wise operations, which rely on the matrix computation for efficient parallelization. As a result, matrices together with higher order variations are easily the most popular mathematical objects for keeping track of data during computation.

In this dissertation, a majority of matrices are this type, even though they are not always the focus of study. For example, multi-output Gaussian process regression uses object-feature matrices for both input and output. In topic modeling, the topic-word matrix is a latent matrix of this type, and plays an crucial role in interpreting the model. Finally, the main matrix of interest in Chapter 6 is formed by having electron orbital pairs as objects and real-space discretization as features.

Matrices for Correlations This type of matrix describes the pairwise relationship of objects rather than object-feature relationships. Examples in this dissertation include the co-occurrence matrix in Chapter 5 that indicates the probability of two objects occurring together in a document, and the kernel matrices in Chapter 4 that measures the covariance between two points for a Gaussian process. The former can be derived from

the document-object matrix, which suggests this type of data matrix can sometimes be considered as higher-order information extracted from the object-feature matrices. The correlation between objects has various benefits and is widely used in statistical models. In our case, it leads to uncertainty quantification, a theoretical optimality guarantee, and algorithmic robustness.

Matrices for Dynamics The last type of data matrix is generally associated with linear time-invariant systems. These matrices appear in Chapter 6 both explicitly as the discretization of the Laplace operator, and implicitly as integral operators—e.g., the Hartree-Fock operator. Here, we focus on exploiting structural information to compress these matrices and gain computational efficiency. On the other hand, in Chapter 3 the graph Laplacian and the random walk matrix describe the dynamics on the original graph, which we use to reveal structural properties. There is a deep connection between the dynamics on an object and its structure. In general, we can leverage our knowledge of one to gain valuable insights into the other.

2.2 Functions of Matrices

The main mathematical problems in both Chapters 3 and 4 can be simplified to estimating key quantities for functions of matrices. In this dissertation, a matrix function $f(A)$ is produced by a spectral mapping of f on a matrix $A \in \mathbb{R}^{N\times N}$. Higham [86] gives a rigorous and general definition of $f(A)$, which only requires f to be *defined*[1] on the spectrum of A. However, we mostly work with a special case, where A is Hermitian and f is analytic over the spectrum of A. Given the eigendecomposition $A = V^T \Lambda V$ where V

[1]Defined is a formal term here. Given a matrix A with distinct eigenvalues $\lambda_1, \cdots, \lambda_s$ and n_i the size of the largest Jordan block for λ_i, the function f is defined on the spectrum of A if $f^{(j)}(\lambda_i)$ exists for $j = 0 : n_i - 1$ and $i = 1 : s$. Please refer to [86, Definition 1.1] for a detailed discussion.

is orthogonal and $\Lambda = \mathrm{diag}(\lambda_1, \cdots, \lambda_N)$ is diagonal, the spectral mapping theorem tells us

$$f(A) = f(V^T \Lambda V) = V^T f(\Lambda) V = V^T \,\mathrm{diag}(f(\lambda_1), \cdots, f(\lambda_N)) V \,. \tag{2.1}$$

Numerical methods for computing functions of matrices are important in many applications, such as differential equations, Markov models, and network science. The straightforward approach in Eq. (2.1) takes an $O(N^3)$ eigendecomposition, which is not always feasible. Many tailored algorithms have been developed for a few specific but prevalent functions, like the matrix square root and matrix exponential [86, Chapter 6 & 10]. In a remarkable work, Moler and Van Loan [138] presented nineteen ways of computing the matrix exponential, comparing these in terms of both efficiency and stability.

Given the context in this dissertation, we adopt two methods for evaluating functions of matrices. The first is a polynomial expansion method in the Chebyshev polynomial basis. The Chebyshev polynomials $T_m(x)$ can be defined with the recurrence:

$$T_0(x) = 1, \quad T_1(x) = x, \quad T_{m+1}(x) = 2xT_m(x) - T_{m-1}(x) \,. \tag{2.2}$$

The corresponding expansion is simply $f(x) \approx \tilde{f}(x) = \sum_{m=0}^{M} c_m T_m(x)$. For any smooth function f, the approximation error decays exponentially with the number of terms in the expansion [173]. Many functions we deal with satisfy the smoothness condition, thereby making this method very attractive in our applications. To evaluate the matrix function $f(A)$, we take advantage of the three-term recurrence relation

$$T_{m+1}(A) = 2AT_m(A) - T_{m-1}(A) \,, \tag{2.3}$$

and avoid storing all $T_m(A)$ at once. When we are only interested in applying $f(A)$ to a vector v, we can further reduce the computation to just calculating $T_m(A)v$ for $m = 1, \cdots, M$, and exploit fast matrix-vector products with A if possible.

The second method is Gauss quadrature and Lanczos (GQL) algorithm proposed by Golub and Meurant [74]. It is based on the Lanczos algorithm, which produces the decomposition

$$AZ_M = Z_M\Gamma_M + r_M e_M^T, \tag{2.4}$$

where $Z_M^T Z_M = I_M$, $Z_M^T r_M = 0$, and Γ_M tridiagonal. If A is numerically low-rank, or has relatively few distinct eigenvalues, it will only take a small number of Lanczos iterations to obtain the approximation $A \approx Z_M\Gamma_M Z_M^T$. Afterwards, we can quickly compute $f(\Gamma_M)$ through eigendecomposition since Γ_M is M-by-M and tridiagonal. The final approximation is $f(A) \approx Z_M f(\Gamma_M) Z_M^T$.

Both methods we introduce here have clear advantages and drawbacks. The Chebyshev expansion method has strong theoretical guarantees on the convergence rate for f with some degree of smoothness over the spectrum of A. Many terms may be required to accurately represent less smooth functions. On the other hand, GQL is well suited for matrices with a few distinct eigenvalues, which allows the Lanczos algorithm to converge quickly. However, it requires extra computation to ensure numerical stability in floating point arithmetic. Therefore, it is crucial for us to determine the numerical characteristics of the matrices we are interested in, and choose the appropriate method for each application.

2.3 Stochastic Estimation

Along with approximation of matrix functions, stochastic estimation is one of the key tools in this dissertation. Given a linear operator $A \in \mathbb{R}^{N\times N}$, often in implicit form, stochastic estimators allow us to efficiently extract the trace and diagonal by probing the operator with random vectors.

The stochastic trace estimator was first proposed by Hutchinson [96], who used the technique to approximate the trace of the influence matrix for Laplacian smoothing splines. In his case, applying influence matrix A to a vector requires solving an expensive smoothing spline; obtaining all diagonal elements exactly — by calculating $e_i^T A e_i$ for $i = 1, \cdots, N$ and e_i the i-th column of the identity matrix — has a significant cost. Instead, Hutchinson applied A to a set of independent probe vectors z such that z_i's are i.i.d. with mean 0 and variance 1. Consequently,

$$\mathbb{E}[z^T A z] = \sum_{i,j} A_{ij}\, \mathbb{E}[z_i z_j] = \mathrm{tr}(A)\,. \tag{2.5}$$

Choosing N_z independent probe vectors Z_j, we obtain the unbiased estimator

$$\mathrm{tr}(A) = \mathbb{E}[z^T A z] \approx \frac{1}{N_z} \sum_{j=1}^{N_z} Z_j^T A Z_j\,. \tag{2.6}$$

Hutchinson also showed that sampling Z_i from the Bernoulli distribution leads to minimal variance on the estimated trace.

Since then, Avron and Toledo [13] reviewed many possible choices of probes for Eqs. (2.5) and (2.7); another common choice is vectors with independent standard normal entries. Bekas et al. [21] extended the idea for diagonal estimation by observing

$$\mathbb{E}[z \odot Az] = \mathrm{diag}(A)\,, \tag{2.7}$$

where $\odot$ represents the Hadamard (elementwise) product. Let $Z \in \mathbb{R}^{N \times N_z}$ so the columns are independent probe vectors. If we inspect the k-th diagonal element,

$$A_{kk} = \mathbb{E}[z_k (Az)_k] = \frac{1}{N_z} \sum_i A_{ki} \sum_{j=1}^{N_z} Z_{kj} Z_{ij} \tag{2.8}$$

Based on Eq. (2.8), Bekas et al. [21] proposed the exact condition for the diagonal estimator.

Theorem 2.1 (Exact Condition for Diagonal Estimatior)**.** *Let* $Z \in \mathbb{R}^{N \times N_z}$ *be the matrix whose columns are probe vectors of the diagonal estimator in Eq.* (2.8)*. Use* Z_{i*} *to denote the i-th row of Z. The estimation of* A_{kk} *is exact if* $\left\|Z_{k*}\right\| = 1$*, and* $Z_{k*}^T Z_{i*} = 0$ *whenever* $H_{ki} \neq 0$ *for* $i \neq k$*.*

The direct consequence of Theorem 2.1 is that we can design a specific set of probe vectors to satisfy the exact condition by solving a graph coloring problem. Treating A as the adjacency matrix of a graph, and the non-zero entries as edges, a graph coloring partitions the nodes into disjoint subsets for which no edge exists between each other. If we assign each subset a row vector chosen from an orthonormal set, the resulting Z will yield the exact diagonal through the same calculation in Eq. (2.8). Although obtaining an optimal coloring scheme is NP-hard, there exists a greedy coloring algorithm using breadth-first-search that can produce reasonable results for some applications [10, Theorem 28.33].

Finally, we also attempted to use a control variate to improve the stochastic estimation. In Chapter 4, where the goal is to estimate the predictive variance, we take the more accurate results from pivoted Cholesky approximation on a few data points as a control variate. Despite little reduction in variance, we are able to obtain an unbiased estimator.

2.4 Interpretable Low-Rank Approximations

Low-rank structure of matrices plays an essential role in many of our methods. If a matrix $A \in \mathbb{R}^{N \times N}$ has rank k, it can be decomposed into a product of two smaller matrices

$W \in \mathbb{R}^{N\times k}$ and $H \in \mathbb{R}^{k\times N}$:

$$A = WH\,. \tag{2.9}$$

Hence, we are able to efficiently store A and multiply it with a vector in $O(Nk)$ rather than $O(N^2)$. Furthermore, the columns of matrix W provide us useful insight about the data, since they form a basis that span the original columns of A. Low-rank approximation is one of the most well-studied problems in linear algebra, and it is widely used in data science and machine learning. High-dimensional data matrices encountered in these fields commonly have a low numerical rank. Correspondingly, researchers have proposed low-rank models for a wide variety of applications, such as principal component analysis, natural language processing, and recommender system. Udell and Townsend [176] formalized this notion by showing a nice latent variable model produces data that can be well approximated by low-rank matrices.

Among the extensive literature on computing and exploiting low-rank structures, there is a growing interest in interpretability. Low-rank approximation via the truncated singular value decomposition optimally approximates the matrix, but often fails to preserve intrinsic properties of datasets, e.g., non-negativity, sparsity, and convexity. Thus, many constrained low-rank approximation algorithms have been developed that retain these properties, and allow us to interpret the output in a natural setting. A prominent example is the non-negative matrix approximation, for which W and H are required to be element-wise non-negative. Lee and Seung [110] applied it to images of faces, and the produced factors can be visualized as components of facial structures. In Chapter 5, the topic model is also a non-negative low-rank decomposition, and we are able to analyze each topic as a probability distribution over words.

One attractive approach to interpretable low-rank approximations forces W to consist of columns from A. This allows us to describe all data points in terms of a small

subset, maintaining the fundamental characteristics. The CUR matrix decomposition, a well-known example of this type, is often preferred over singular value decomposition in certain applications because of its efficiency and interpretability. The biggest challenge here is to select a good subset of data points to obtain an accurate representation. The column subset selection problem is also a heavily discussed topic in numerical linear algebra [28, 47]. One of the most popular building block for solving it is QR factorization with column pivoting (QRCP). The column pivoting procedure generally gives QR factorization the rank-revealing property. Combined with randomized sampling techniques, QRCP is able to effectively select columns that lead to near-optimal low-rank approximations. Both Chapters 5 and 6 tackle the column subset selection problem, but from different perspectives. Chapter 5 develops a scalable pre-processing framework to overcome the robustness issue for QRCP on a noisy dataset. Chapter 6 exploits the physical attributes in the system to replace pivots from QRCP with centroids from the weighted K-means algorithm. The resulting decomposition is much more efficient without any loss in accuracy.

CHAPTER 3

NETWORK DENSITY OF STATES

LIST OF SYMBOLS

A	Adjacency matrix	Page 17
D	Degree matrix	Page 17
G	Graph	Page 17
H	Hermitian matrix	Page 18
J_m	Jackson smoothing factors	Page 26
K_σ	Mollifier	Page 19
L	Laplacian matrix	Page 17
N	Number of nodes	Page 16
N_z	Number of probe vectors	Page 22
P	Filtering matrix	Page 26
T_m	Chebyshev polynomials	Page 21
T_m^*	Dual Chebyshev polynomials	Page 21
V	Set of vertices	Page 17
W_1	Wasserstein distance	Page 19
Z	Matrix with independent probe vectors as columns	Page 21
$\mathbb{E}$	Expected value	Page 21
δ_{mn}	Kronecker delta function	Page 21
λ_i	Eigenvalues	Page 18
$\mu(\lambda)$	Density of states	Page 18
$\mu(\lambda; u)$	Local density of states	Page 19
$\underline{\mu}_k(\lambda)$	Pointwise density of states	Page 19
$\overline{A}$	Normalized adjacency matrix	Page 18
$\overline{L}$	Normalized Laplacian matrix	Page 18
diag	Diagonal of a matrix	Page 21
tr	Trace of a matrix	Page 18
c_m	Chebyshev moments	Page 26
d_m	Dual Chebyshev moments	Page 21
q_i	Eigenvectors	Page 18
z	Probe vector	Page 21

Spectral analysis connects graph structure to the eigenvalues and eigenvectors of associated matrices. Much of spectral graph theory descends directly from spectral geometry, the study of differentiable manifolds through the spectra of associated differential operators. But the translation from spectral geometry to spectral graph theory has largely focused on results involving only a few extreme eigenvalues and their associated eigenvalues. Unlike in geometry, the study of graphs through the overall distribution of eigenvalues — the *spectral density* — is largely limited to simple random graph models. The interior of the spectrum of real-world graphs remains largely unexplored, difficult to compute and to interpret.

In this chapter, we delve into the heart of spectral densities of real-world graphs. We borrow tools developed in condensed matter physics, and add novel adaptations to handle the spectral signatures of common graph motifs. The resulting methods are highly efficient, as we illustrate by computing spectral densities for graphs with over a billion edges on a single compute node. Beyond providing visually compelling fingerprints of graphs, we show how the estimation of spectral densities facilitates the computation of many common centrality measures, and use spectral densities to estimate meaningful information about graph structure that cannot be inferred from the extremal eigenpairs alone.

3.1 Introduction

Spectral theory is a powerful analysis tool in graph theory [44, 38, 37], geometry [34], and physics [98]. One follows the same steps in each setting:

- Identify an object of interest, such as a graph or manifold;

- Associate the object with a matrix or operator, often the generator of a linear dynamical system or the Hessian of a quadratic form over functions on the object;
- Connect spectral properties of the matrix or operator to structural properties of the original object.

In each case, the *complete* spectral decomposition is enough to recover the original object; the interesting results relate structure to *partial* spectral information.

Many spectral methods use extreme eigenvalues and associated eigenvectors. These are easy to compute by standard methods, and are easy to interpret in terms of the asymptotic behavior of dynamical systems or the solutions to quadratic optimization problems with quadratic constraints. Several network centrality measures, such as PageRank [147], are expressed via the stationary vectors of transition matrices, and the rate of convergence to stationarity is bounded via the second-largest eigenvalue. In geometry and graph theory, Cheeger's inequality relates the second-smallest eigenvalue of a Laplacian or Laplace-Beltrami operator to the size of the smallest bisecting cut [35, 137]; in the graph setting, the associated eigenvector (the Fiedler vector) is the basis for spectral algorithms for graph partitioning [152]. Spectral algorithms for graph coordinates and clustering use the first few eigenvectors of a transition matrix or (normalized) adjacency or Laplacian [23, 144]. For a survey of such approaches in network science, we refer to [37].

Mark Kac popularized an alternate approach to spectral analysis in an expository article [100] in which he asked whether one can determine the shape of a physical object (Kac used a drum as an example) given the spectrum of the Laplace operator; that is, can one "hear" the shape of a drum? One can ask a similar question in graph theory: can one uniquely determine the structure of a network from the spectrum of the Laplacian or another related matrix? Though the answer is negative in both cases [75, 44], the

spectrum is enormously informative even without eigenvector information. Unlike the extreme eigenvalues and vectors, eigenvalues deep in the spectrum are difficult to compute and to interpret, but the overall distribution of eigenvalues — known as the spectral density or density of states — provides valuable structural information. For example, knowing the spectrum of a graph adjacency matrix is equivalent to knowing $\mathrm{tr}(A^k)$, the number of closed walks of any given length k. In some cases, one wants *local* spectral densities in which the eigenvalues also have positive weights associated with a location. Following Kac, this would give us not only the frequencies of a drum, but also amplitudes based on where the drum is struck. In a graph setting, the local spectral density of an adjacency matrix at node j is equivalent to knowing $(A^k)_{jj}$, the number of closed walks of any given length k that begin and end at the node.

Unfortunately, the analysis of spectral densities of networks has been limited by a lack of scalable algorithms. While the normalized Laplacian spectra of Erdős-Rényi random graphs have an approximately semicircular distribution [186], and the spectral distributions for other popular scale-free and small-world random graph models are also known [61], there has been relatively little work on computing spectral densities of large "real-world" networks. Obtaining the full eigendecomposition is $O(N^3)$ for a graph with N nodes, which is prohibitive for graphs of more than a few thousand nodes. In prior work, researchers have employed methods, such as thick-restart Lanczos, that still do not scale to very large graphs [61], or heuristic approximations with no convergence analysis [16]. It is only recently that clever computational methods were developed simply to *test* for hypothesized power laws in the spectra of large real-world matrices by computing only *part* of the spectrum [57].

In this chapter, we show how methods used to study densities of states in condensed matter physics [182] can be used to study spectral densities in networks. We study these

methods for both the *global* density of states and for *local* densities of states weighted by specific eigenvector components. We adapt these methods to take advantage of graph-specific structure not present in most physical systems, and analyze the stability of the spectral density to perturbations as well as the convergence of our computational methods. Our methods are remarkably efficient, as we illustrate by computing densities for graphs with billions of edges and tens of millions of nodes on a single cloud compute node. We use our methods for computing these densities to create compelling visual fingerprints that summarize a graph. We also show how the density of states reveals graph properties that are not evident from the extremal eigenvalues and eigenvectors alone, and use it as a tool for fast computation of standard measures of graph connectivity and node centrality. This opens the door for the use of complete spectral information as a tool in large-scale network analysis.

3.2 Background

3.2.1 Graph Operators and Eigenvalues

We consider weighted, undirected graphs $G = (V, E)$ with vertices $V = \{v_1, \cdots, v_N\}$ and edges $E \subseteq V \times V$. The weighted adjacency matrix $A \in \mathbb{R}^{N \times N}$ has entries $a_{ij} > 0$ to give the weight of an edge $(i, j) \in E$ and $a_{ij} = 0$ otherwise. The degree matrix $D \in \mathbb{R}^{N \times N}$ is the diagonal matrix of weighted node degrees, i.e. $D_{ii} = \sum_j a_{ij}$. Several of the matrices in spectral graph theory are defined in terms of D and A. We describe a few of these below, along with their connections to other research areas. For each operator, we let $\lambda_1 \leq \ldots \leq \lambda_N$ denotes the eigenvalues in ascending order.

Adjacency Matrix: A Many studies on the spectrum of A originate from random matrix theory where A represents a random graph model. In these cases, the limiting behavior of eigenvalues as $N \to \infty$ is of particular interest. Besides the growth of extremal eigenvalues [38], Wigner's semicircular law is the most renowned result about the spectral distribution of the adjacency matrix [186]. When the edges are i.i.d. random variables with bounded moments, the density of eigenvalues within a range converges to a semicircular distribution. One famous graph model of this type is the Erdős-Rényi graph, where $a_{ij} = a_{ji} = 1$ with probability $p < 1$, and 0 with probability $1 - p$. Farkas et al. [61] has extended the semicircular law by investigating the spectrum of scale-free and small-world random graph models. They show the spectra of these random graph models relate to geometric characteristics such as the number of cycles and the degree distribution.

Laplacian Matrix: $L = D - A$ The Laplace operator arises naturally from the study of dynamics in both spectral geometry and spectral graph theory. The continuous Laplace operator and its generalizations are central to the description of physical systems including heat diffusion [132], wave propagation [114], and quantum mechanics [56]. It has infinitely many non-negative eigenvalues, and Weyl's law [184] relates their asymptotic distribution to the volume and dimension of the manifold. On the other hand, the discrete Laplace matrix appears in the formulation of graph partitioning problems. If $f \in \{\pm 1\}^N$ is an indicator vector for a partition $V = V_+ \cup V_-$, then $f^T L f/4$ is the number of edges between V_+ and V_-, also known as the cut size. L is a positive-semidefinite matrix with the vector of all ones as a null vector. The eigenvalue λ_2, called the *algebraic connectivity*, bounds from below the smallest bisecting cut size; $\lambda_2 = 0$ if and only if the graph is disconnected. In addition, eigenvalues of L also appear in bounds for vertex connectivity (λ_2) [45], minimal bisection (λ_2) [50], and maximum cut (λ_N) [174].

Normalized Laplacian Matrix: $\overline{L} = I - D^{-1/2}AD^{-1/2}$ We will also mention the normalized adjacency matrix $\overline{A} = D^{-1/2}AD^{-1/2}$ and graph random walk matrix $P = D^{-1}A$ here, because these matrices have the same eigenvalues as $\bar{L}$ up to a shift. The connection to some of the most influential results in spectral geometry is established in terms of eigenvalues and eigenvectors of normalized Laplacian. A prominent example is the extension of Cheeger's inequality to the discrete case, which relates the set of smallest conductance $h(G)$ (the Cheeger constant) to the second smallest eigenvalue of the normalized Laplacian, $\lambda_2(\overline{L})$ [139]:

$$\lambda_2(\overline{L})/2 \leq h(G) = \min_{S \subset V} \frac{|\{(i,j) \in E, i \in S, j \notin S\}|}{\min(\text{vol}(S), \text{vol}(V \backslash S))} \leq \sqrt{2\lambda_2(\overline{L})}, \tag{3.1}$$

where $\text{vol}(S) = \sum_{i \in S} \sum_{j=1}^{N} a_{ij}$. Cheeger's inequality offers crucial insights and powerful techniques for understanding popular spectral graph algorithms for partitioning [133] and clustering [144]. It also plays a key role in analyzing the mixing time of Markov chains and random walks on a graph [136, 167]. For all these problems, extremal eigenvalues again emerge from relevant optimization formulations.

3.2.2 Spectral Density (Density of States — DOS)

Let $H \in \mathbb{R}^{N \times N}$ be any symmetric graph matrix with an eigendecomposition $H = Q\Lambda Q^T$, where $\Lambda = \text{diag}(\lambda_1, \cdots, \lambda_N)$ and $Q = [q_1, \cdots, q_N]$ is orthogonal. The spectral density induced by H is the generalized function

$$\mu(\lambda) = \frac{1}{N} \sum_{i=1}^{N} \delta(\lambda - \lambda_i), \quad \int f(\lambda)\mu(\lambda) = \text{tr}(f(H)) \tag{3.2}$$

where δ is the Dirac delta function and f is any analytic test function. The spectral density μ is also referred to as the *density of states* (DOS) in the condensed matter physics literature [182], as it describes the number of states at different energy levels.

For any vector $u \in \mathbb{R}^N$, the local density of states (LDOS) is

$$\mu(\lambda; u) = \sum_{i=1}^{N} |u^T q_i|^2 \delta(\lambda - \lambda_i), \qquad \int f(\lambda)\mu(\lambda; u) = u^T f(H) u. \tag{3.3}$$

Most of the time, we are interested in the case $u = e_k$ where e_k is the kth standard basis vector—this provides the spectral information about a particular node. We will write $\mu_k(\lambda) = \mu(\lambda; e_k)$ for the pointwise density of states (PDOS) for node v_k. It is noteworthy $|e_k^T q_i| = |q_i(k)|$ gives the magnitude of the weight for v_k in the i-th eigenvector, thereby the set of $\{\mu_k\}$ encodes the entire spectral information of the graph up to sign differences. These concepts can be easily extended to directed graphs with asymmetric matrices, for which the eigenvalues are replaced by singular values, and eigenvectors by left/right singular vectors.

Naively, to obtain the DOS and LDOS requires computing all eigenvalues and eigenvectors for an N-by-N matrix, which is infeasible for large graphs. Therefore, we turn to algorithms that approximate these densities. Since the DOS is a generalized function, it is important we specify how the estimation is evaluated. One choice is to treat μ (or μ_k) as a distribution, and measure its approximation error with respect to a chosen function space $\mathcal{L}$. For example, when $\mathcal{L}$ is the set of Lipschitz continuous functions taking the value 0 at 0, the error for estimated $\widetilde{\mu}$ is in the Wasserstein distance (a.k.a. earth-mover distance) [101]

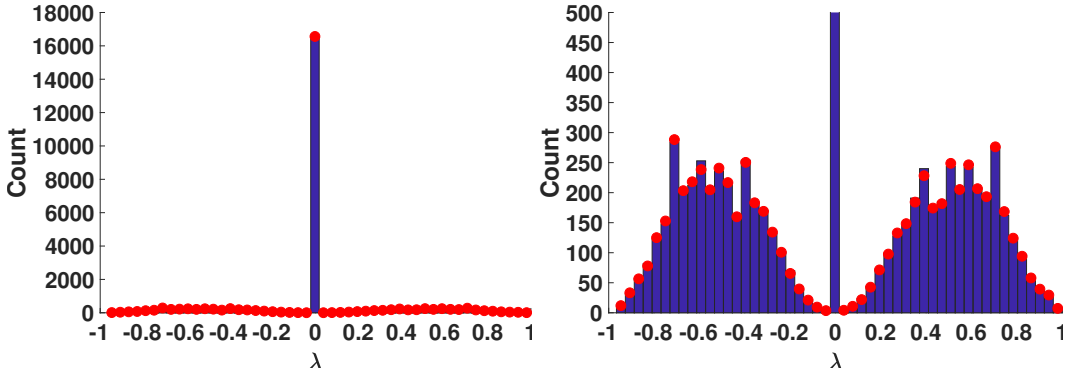
$$W_1(\mu, \widetilde{\mu}) = \sup\left\{ \int (\mu(\lambda) - \widetilde{\mu}(\lambda)) f(\lambda) d\lambda : \mathrm{Lip}(f) \le 1 \right\}. \tag{3.4}$$

This notion is particularly useful when μ is integrated against in applications such as computing centrality measures.

On the other hand, we can regularize μ with a mollifier K_σ (i.e., a smooth approximation of the identity function):

$$(K_\sigma * \mu)(\lambda) = \int_{\mathbb{R}} \sigma^{-1} K\left(\frac{\lambda - \nu}{\sigma}\right) \mu(\nu) d\nu \tag{3.5}$$

A simplified approach is numerically integrating μ over small intervals of equal size to generate a spectral histogram. The advantage is the error is now easily measured and visualized in the L_∞ norm. For example, Figure 3.1 shows the exact and approximated spectral histogram for the normalized adjacency matrix of an Internet topology.



(a) Spectral Histogram (b) Zoom-in View

Figure 3.1: Spectral histogram for the normalized adjacency matrix for the CAIDA autonomous systems graph [95], an Internet topology with $22,965$ nodes and $47,193$ edges. Blue bars are the real spectrum, and red points are the approximated heights. (a) contains high multiplicity around eigenvalue 0, so (b) zooms in to height between $[0, 500]$.

3.3 Methods

The density of states plays a significant role in understanding electronic band structure in solid state physics, and so several methods have been proposed in that literature to estimate spectral densities. We review two such methods: the kernel polynomial method (KPM) which involves a polynomial expansion of the DOS/LDOS, and the Gauss Quadrature via Lanczos iteration (GQL). These methods have not previously been applied in the network setting, though Cohen-Steiner et al. [39] have independently invented an approach similar to KPM for the global DOS alone, albeit using a less numerically stable polynomial basis (the power basis associated with random walks). We

then introduce a new direct nested dissection method for LDOS, as well as new graph-specific modifications to improve the convergence of the KPM and GQL approaches.

Throughout this section, H denotes any symmetric matrix.

3.3.1 Kernel Polynomial Method (KPM)

The Kernel Polynomial Method (KPM) [182] approximates the spectral density through an expansion in the dual basis of an orthogonal polynomial basis. Traditionally, the Chebyshev basis $\{T_m\}$ is used because of its connection to the best polynomial interpolation. Chebyshev approximation requires the spectrum to be supported on the interval $[-1, 1]$ for numerical stability. However, this condition can be satisfied by any graph matrix after shifting and rescaling:

$$\widetilde{H} = \frac{2H - (\lambda_{\max}(H) + \lambda_{\min}(H))}{\lambda_{\max}(H) - \lambda_{\min}(H)}\,. \tag{3.6}$$

We can compute these extremal eigenvalues efficiently for our sparse matrix H, so the pre-computation is not an issue [148].

As defined in Eq. (2.2), the Chebyshev polynomials $T_m(x)$ satisfy the recurrence

$$T_0(x) = 1,\ T_1(x) = x,\ T_{m+1}(x) = 2xT_m(x) - T_{m-1}(x)\,.$$

They are orthogonal with respect to $w(x) = 2/\left[(1 + \delta_{0n})\pi\sqrt{1 - x^2}\right]$:

$$\int_{-1}^{1} w(x)T_m(x)T_n(x)dx = \delta_{mn}\,. \tag{3.7}$$

(Here and elsewhere, δ_{ij} is the Kronecker delta: 1 if $i = j$ and 0 otherwise.) Therefore, $T^*_m(x) = w(x)T_m(x)$ also forms the dual Chebyshev basis. Using Eq. 3.7, we can expand

our DOS $\mu(\lambda)$ as

$$\mu(\lambda) = \sum_{m=0}^{\infty} d_m T_m^*(\lambda)\,, \tag{3.8}$$

$$d_m = \int_{-1}^{1} T_m(\lambda)\mu(\lambda)d\lambda = \frac{1}{N}\sum_{i=1}^{N} T_m(\lambda_i) = \frac{1}{N}\,\mathrm{tr}(T_m(H))\,. \tag{3.9}$$

Here, $T_m(H)$ is the m-th Chebyshev polynomial of the matrix H. The last equality comes from the spectral mapping theorem, which says that taking a polynomial of H maps the eigenvalues by the same polynomial. Similarly, we express the PDOS $\mu_k(\lambda)$ as

$$d_{mk} = \int_{-1}^{1} T_m(\lambda)\mu_k(\lambda)d\lambda = \sum_{i=1}^{N} |q_i(k)|^2 T_m(\lambda_i) = T_m(H)_{kk}\,. \tag{3.10}$$

We want to efficiently extract the diagonal elements of the matrices $\{T_m(H)\}$ without forming them explicitly; the key idea is to apply the stochastic trace/diagonal estimation, proposed by Hutchinson [96] and Bekas et al. [21]. We have described the details in Section 2.3. Using the Chebyshev recurrence (Eq. (2.2)), we can compute the sequence $T_m(H)z$ for each probe at a cost of one matrix-vector product per term, for a total cost of $O(|E|\,N_z)$ time per moment $T_m(H)$.

$$T_{m+1}(H)z = 2HT_m(H)z - T_{m-1}(H)z\,. \tag{3.11}$$

In practice, we only use a finite number of moments rather than an infinite expansion. The number of moments required depends on the convergence rate of the Chebyshev approximation for the class of functions DOS/LDOS is integrated with. For example, the approximation error decays exponentially for test functions that are smooth over the spectrum [173], so only a few moments are needed. On the other hand, such truncation leads to Gibbs oscillations that cause error in the interpolation [173]. However, to a large extent, we can use smoothing techniques such as Jackson damping to resolve this issue [97] (we will formalize this in Theorem 3.1).

3.3.2 Gauss Quadrature and Lanczos (GQL)

This method builds of the Gauss Quadrature and Lanczos (GQL) algorithm we introduced in Section 2.2. Using the same stochastic estimation from Section 3.3.1, we can also apply GQL to compute DOS.

For a starting vector z and graph matrix H, Lanczos iterations after M steps produce a decomposition

$$HZ_M = Z_M \Gamma_M + r_M e_M^T \,,$$

where $Z_M^T Z_M = I_M$, $Z_M^T r_M = 0$, and Γ_M tridiagonal. GQL approximates $z^T f(H) z$ with $\|z\|^2 e_1^T f(T_M) e_1$, implying

$$z^T f(H) z = \sum_{i=1}^{N} \left| z^T q_i \right|^2 f(\lambda_i) \approx \|z\|^2 \sum_{i=1}^{M} \left| p_{i1} \right|^2 f(\tau_i) \,, \tag{3.12}$$

where $(\tau_1, p_1) \cdots, (\tau_M, p_M)$ are the eigenpairs of Γ_M. Consequently,

$$\|z\|^2 \sum_{i=1}^{M} \left| p_{i1} \right|^2 \delta(\lambda - \tau_i) \tag{3.13}$$

approximates the LDOS $\mu(\lambda; z)$.

Building upon the stochastic estimation idea and the invariance of probe vectors under orthogonal transformation, we have

$$\mathbb{E}[\mu(\lambda; z)] = \sum_{i=1}^{N} \delta(\lambda - \lambda_i) = N\mu(\lambda) \,. \tag{3.14}$$

Hence

$$\mu(\lambda) \approx \sum_{i=1}^{M} |p_{i1}|^2 \delta(\lambda - \tau_i) \,. \tag{3.15}$$

The approximate generalized function is exact when applied to polynomials of degree $\leq 2M + 1$. Furthermore, if we let $z = e_k$ then GQL also provides an estimation for the PDOS $\mu_k(\lambda)$. Estimation from GQL can also be converted to Chebyshev moments if needed.

3.3.3 Nested Dissection (ND)

The estimation error via Monte Carlo method intrinsically decays at the rate $O(1/\sqrt{N_z})$, where N_z is the number of random probing vectors. Hence, we have to tolerate the higher variance when increasing the number of probe vectors becomes too expensive. This is particularly problematic when we try to compute the PDOS for all nodes using the stochastic diagonal estimator. Therefore, we introduce an alternative divide-and-conquer method, which computes more accurate PDOS for any set of nodes at a cost comparable to the stochastic approach in practice.

Suppose the graph can be partitioned into two subgraphs by removal of a small vertex separator. Permuting the vertices so that the two partitions appear first, followed by the separator vertices. Up to vertex permutations, we can rewrite H in block form as

$$H = \begin{bmatrix} H_{11} & 0 & H_{13} \\ 0 & H_{22} & H_{23} \\ H_{13}^T & H_{23}^T & H_{33} \end{bmatrix}, \tag{3.16}$$

where the indices indicate the groups identities. Leveraging this structure, we can update the recurrence relation for Chebyshev polynomials to become

$$T_{m+1}(H)_{11} = 2H_{11}T_m(H)_{11} - T_{m-1}(H)_{11} + 2H_{13}T_m(H)_{31}\,. \tag{3.17}$$

Recursing on the partitioning will lead to a nested dissection, after which we will use direct computation on sufficiently small sub-blocks. We denote the indexing of each partition with $I_p^{(t)} = I_s^{(t)} \bigcup I_\ell^{(t)} \bigcup I_r^{(t)}$, which represents all nodes in the current partition, the separators, and two sub-partitions, respectively. For the separators, Eq. (3.17) leads

to

$$T_{m+1}(H)(I_p^{(t)}, I_s^{(t)}) = 2H(I_p^{(t)}, I_p^{(t)})T_m(H)(I_p^{(t)}, I_s^{(t)}) - T_{m-1}(H)(I_p^{(t)}, I_s^{(t)}) + 2\sum_{t' \in S_t} H(I_p^{(t)}, I_s^{(t')})T_m(H)(I_s^{(t')}, I_s^{(t)}), \tag{3.18}$$

where S_t is the path from partition t to the root; and for the leaf blocks, $I_s^{(t)} = I_p^{(t)}$ in Eq. (3.18). The result is Algorithm 3.1.

Algorithm 3.1: Nested Dissection for PDOS Approximation

Input: Symmetric graph matrix H with eigenvalues in $[-1, 1]$
Output: $C \in \mathbb{R}^{N \times M}$ where c_{ij} is the j-th Chebyshev moment for i-th node.
begin
  Obtain partitions $\{I_p^{(t)}\}$ in a tree structure through multilevel nested dissection.
  for $m = 1$ ***to*** M **do**
    Traverse partition tree in pre-order:
      Compute the separator columns with Eq. (3.18).
      if $I_p^{(t)}$ *is a leaf block* **then**
        Compute the diagonal entries with equation (3.18).
      end
  end
end

The multilevel nested dissection process itself has a well-established algorithm by Karypis and Kumar, and efficient implementation is available in `METIS` [102]. Note that this approach is only viable when the graph can be partitioned with a separator of small size. Empirically, we observe this assumption to hold for many real-world networks. The biggest advantage of this approach is we can very efficiently obtain PDOS estimation for a subset of nodes with much better accuracy than KPM.

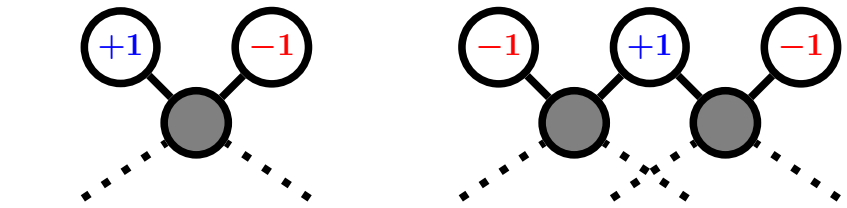
3.3.4 Motif Filtering

In many graphs, there are large spikes around particular eigenvalues; for example, see Fig. 3.1. This phenomenon affects the accuracy of DOS estimation in two ways. First, the singularity-like behavior means we need many more moments to obtain a good approximation in polynomial basis. Secondly, due to the equi-oscillation property of Chebyshev approximation, error in irregularities (say, at a point of high concentration in the spectral density), spreads to other parts of the spectrum. This is a problem in our case, as the spectral density of real-world networks are far from uniform.

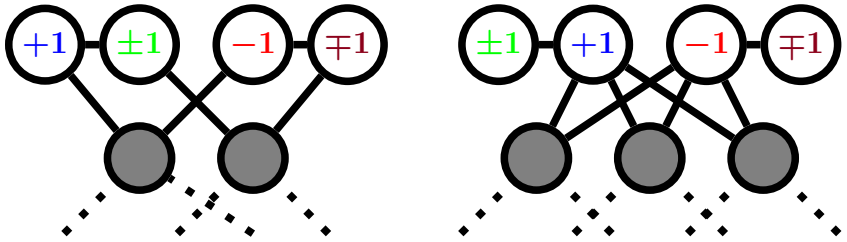
High multiplicity eigenvalues are typically related to local symmetries in a graph. The most prevalent example is two dangling nodes attached to the same neighbor as shown in Fig. 3.2a, which accounts for most eigenvalues around 0 for (normalized) adjacency matrix with a localized eigenvector taking value $+1$ on one node and -1 on the other. In addition, we list a few more motifs in Fig. 3.2 that appear most frequently in real-world graphs. All of them can be associated with specific eigenvalues, and we include the corresponding ones in normalized adjacency matrix for our example.

To detect these motifs in large graphs, we deploy a randomized hashing technique. Given a random vector z, the hashing weight $w = Hz$ encodes all the neighborhood information of each node. To find node copies (left in Figure 3.2a), we seek pairs (i, j) such that $w_i = w_j$; with high probability, this only happens when v_i and v_j share the same neighbors. Similarly, all motifs in Figure 3.2 can be characterized by union and intersection of neighborhood lists.

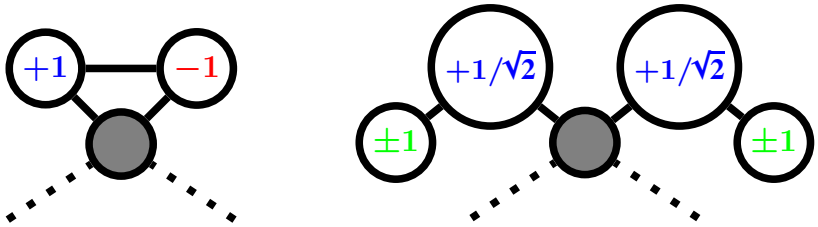
After identifying motifs, we need only approximate the (relatively smooth) density of the remaining spectrum. The eigenvectors associated with these remaining non-motif eigenvalues must be constant across cycles in the canonical decomposition of the associated permutations. Let $P \in \mathbb{R}^{N\times r}$ denote an orthonormal basis for the space of such



(a) $\lambda = 0$



(b) $\lambda = \pm 1/2$



(c) $\lambda = -1/2$ (d) $\lambda = \pm 1/\sqrt{2}$

Figure 3.2: Common motifs (induced subgraphs) in graph data that result in localized spikes in the spectral density. Each motif generates a specific eigenvalue with locally-supported eigenvectors. Here we uses the normalized adjacency matrix to represent the graph, although we can perform the same analysis for the adjacency, Laplacian, or normalized Laplacian (only the eigenvalues would be different). The eigenvectors are supported only on the labeled nodes.

vectors formed from columns of the identity and (normalized) indicators for nodes cyclically permuted by the motif. The matrix $H_r = P^T H P$ then has identical eigenvalues to H, except with all the motif eigenvalues omitted. We may form H_r explicitly, as it has the same sparsity structure as H but with a supernode replacing the nodes in each instance of a motif cycle; or we can achieve the same result by replacing each random probe Z with the projected probe $Z_r = PP^T Z$ at an additional cost of $O(N_{\text{motif}})$ per probe,

where N_{motif} is the number of nodes involved in motifs.

The motif filtering method essentially allows us to isolate the spiky components from the spectrum. As a result, we are able to obtain a more accurate approximation using fewer Chebyshev moments. Fig. 3.3 demonstrates the improvement on the approximation as we procedurally filter out motifs at 0, $-1/3$, $-1/2$, and $-1/4$. The eigenvalue $-1/m$ can be generated by an edge attached to the graph through $m-1$ nodes, similar to motif (3.2c).

3.4 Error Analysis

3.4.1 KPM Approximation Error

This section provides an error bound for our regularized DOS approximation $K_\sigma * \mu$. We will start with the following theorem.

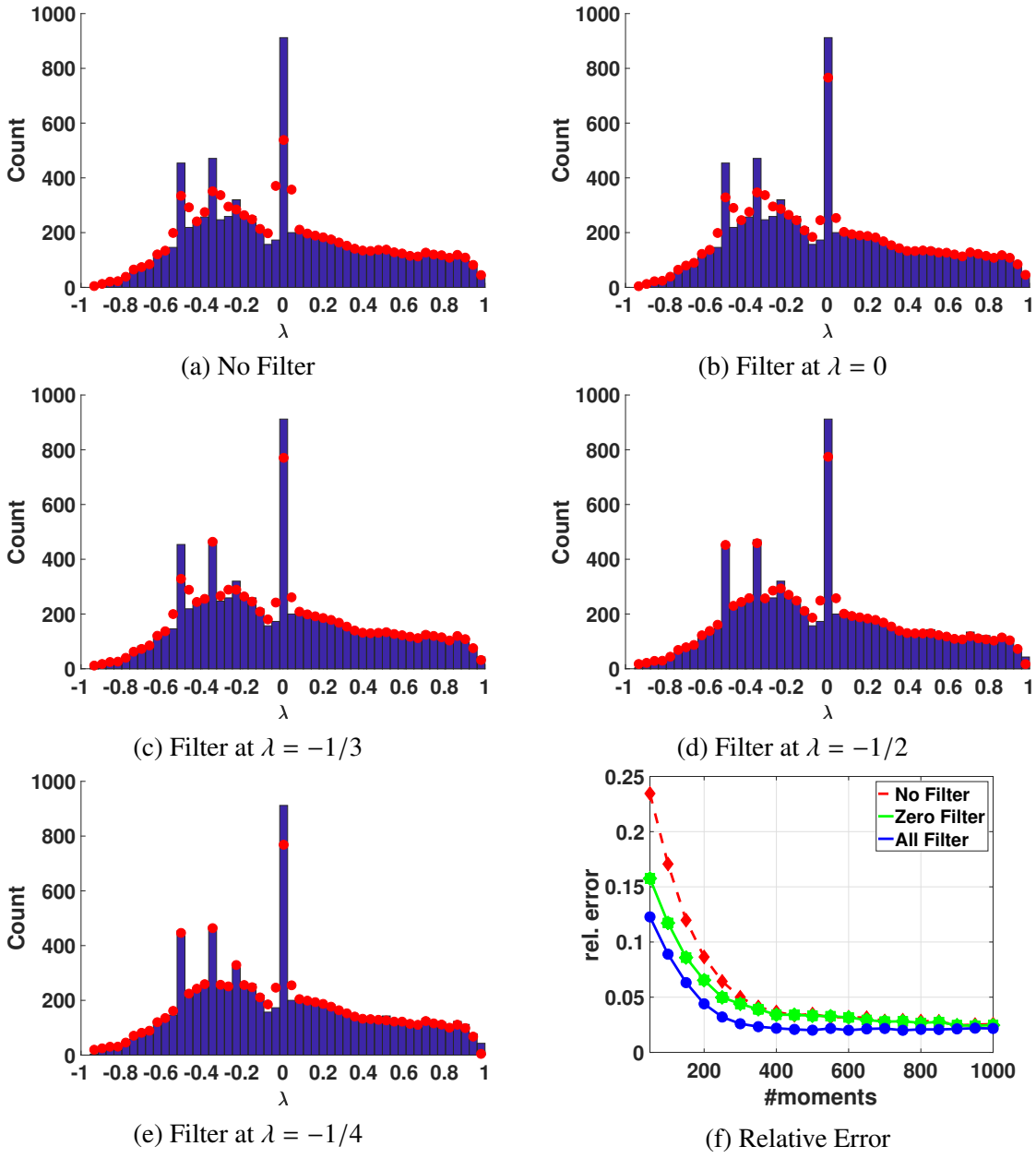
Theorem 3.1 (**Jackson's Theorem [97]**). *If $f : [-1, 1] \to \mathbb{R}$ is Lipschitz continuous with constant L, its best degree M polynomial approximation $\hat{f}^M$ has an L_∞ error of at most $6L/M$. The approximation can be constructed as*

$$\hat{f}^M = \sum_{m=0}^{M} J_m c_m T_m(x)\,, \tag{3.19}$$

where J_m are Jackson smoothing factors and c_m are the Chebyshev coefficients.

We can pick a smooth mollifier K with $\mathrm{Lip}(K) = 1$. For any $\nu \in \mathbb{R}$ and $\lambda \in [-1, 1]$ there exists a degree M polynomial such that

$$\left| K_\sigma(\nu - \lambda) - \widehat{K}_\sigma^M(\nu - \lambda) \right| < \frac{6L}{M\sigma}\,. \tag{3.20}$$



(a) No Filter

(b) Filter at $\lambda = 0$

(c) Filter at $\lambda = -1/3$

(d) Filter at $\lambda = -1/2$

(e) Filter at $\lambda = -1/4$

(f) Relative Error

Figure 3.3: The improvement in accuracy of the spectral histogram approximation on the normalized adjacency matrix for the High Energy Physics Theory (HepTh) Collaboration Network, as we sweep through spectrum and filter out motifs. The graph has 8,638 nodes and 24,816 edges. Blue bars are the real spectrum, and red points are the approximated heights. Fig. 3.3a- 3.3e use 100 moments and 20 probe vectors. Fig. 3.3f shows the relative L_1 error of the spectral histogram when using no filter, filter at $\lambda = 0$, and all filters.

Define $\hat{\mu}^M = \sum_{m=0}^{M} J_m d_m \phi_m$ to be the truncated DOS series,

$$\int_{-1}^{1} \hat{f}^M(\lambda)\mu(\lambda)d\lambda = \int_{-1}^{1} f(\lambda)\hat{\mu}^M(\lambda)d\lambda = \sum_{m=0}^{M} J_m c_m d_m \,. \tag{3.21}$$

Therefore,

$$\begin{aligned}\left\|K_\sigma * (\mu - \hat{\mu}^M)\right\|_\infty &= \max_\nu \left| \int_{-1}^{1} K_\sigma(\nu - \lambda)(\mu(\lambda) - \hat{\mu}^M(\lambda))d\lambda \right| \\ &\le \max_\nu \int_{-1}^{1} \left| K_\sigma(\nu - \lambda) - \widehat{K}_\sigma^M(\nu - \lambda) \right| \mu(\lambda)d\lambda \\ &\le \frac{6L}{M\sigma} \,.\end{aligned}$$

Consider $\tilde{\mu}^M$ to be the degree M approximation from KPM,

$$\left\|K_\sigma * (\mu - \tilde{\mu}^M)\right\|_\infty \le \left\|K_\sigma * (\mu - \hat{\mu}^M)\right\|_\infty + \left\|K_\sigma\right\|_\infty \left\|\hat{\mu}^M - \tilde{\mu}^M\right\|_1 \,. \tag{3.22}$$

If we use a probe z with independent standard normal entries for the trace estimation,

$$\tilde{\mu}(\lambda) = \sum_{i=1}^{N} w_i^2 \delta(\lambda - \lambda_i) \tag{3.23}$$

where $w = Q^T z$ is the weight for z in the eigenbasis. Hence

$$\left\|\hat{\mu}^M - \tilde{\mu}^M\right\|_1 \le \sum_{i=1}^{N} \left|1 - w_i^2\right| \,. \tag{3.24}$$

Finally,

$$\mathbb{E}\left[\left\|K_\sigma * (\mu - \tilde{\mu}^M)\right\|\right] \le \frac{1}{\sigma}\left(\frac{6L}{M} + \|K\|_\infty \mathbb{E}\left[\left|1 - w_1^2\right|\right]\right) \,. \tag{3.25}$$

If we take N_z independent probe vectors, then $N_z w_1^2 \sim \chi^2(N_z)$, which means the expectation decays asymptotically like $\sqrt{2/(\pi N_z)}$.

3.4.2 Perturbation Analysis

In this section, we limit our attention to symmetric graph matrix H. Extracting graph information using DOS, whether as a distribution for functions on a graph or as a direct

feature in the form of spectral moments, requires stability under small perturbations. In the case of removing/adding a few number of nodes/edges, the Cauchy Interlacing Theorem [128] gives a bound on each individual new eigenvalue by the old ones. For example, if we remove $r \ll N$ nodes to get a new graph matrix $\widetilde{H}$, then

$$\lambda_i(H) \leq \lambda_i(\widetilde{H}) \leq \lambda_{i+r}(H) \quad \text{for} \quad i \leq N - r\,. \tag{3.26}$$

However, this bound may not be helpful when the impact of the change is not reflected by its size. Hence, we provide a theorem that relates the Wasserstein distance (see Eq. (3.4)) change and the Frobenius norm of the perturbation. Without loss of generality, we assume the eigenvalues of H lie in $[-1, 1]$ already.

Theorem 3.2. *Suppose $\widetilde{H} = H + \delta H$ is the perturbed graph matrix with spectral density $\tilde{\mu}$, then*

$$W_1(\mu, \tilde{\mu}) \leq \|\delta H\|_F$$

Proof. Let $\mathcal{L}$ be the space of Lipschitz functions with $f(0) = 0$.

$$\begin{aligned} W_1(\mu, \tilde{\mu}) &= \sup_{f \in \mathcal{L}, \mathrm{Lip}(f)=1} \int f(\lambda)(\mu(\lambda) - \tilde{\mu}(\lambda))d\lambda \\ &= \frac{1}{N} \sup_{f \in \mathcal{L}, \mathrm{Lip}(f)=1} \mathrm{tr}(f(H) - f(\widetilde{H})) \\ &\leq \sup_{f \in \mathcal{L}, \mathrm{Lip}(f)=1, \|v\|=1} v^T(f(H) - f(\widetilde{H}))v\,. \end{aligned}$$

By Theorem 3.8 from Higham [86], the perturbation on $f(H)$ is bounded by the Fréchet derivative,

$$\left\|f(H) - f(\widetilde{H})\right\|_2 \leq \mathrm{Lip}(f)\|\delta H\|_F + o(\|\delta H\|_F). \tag{3.27}$$

□

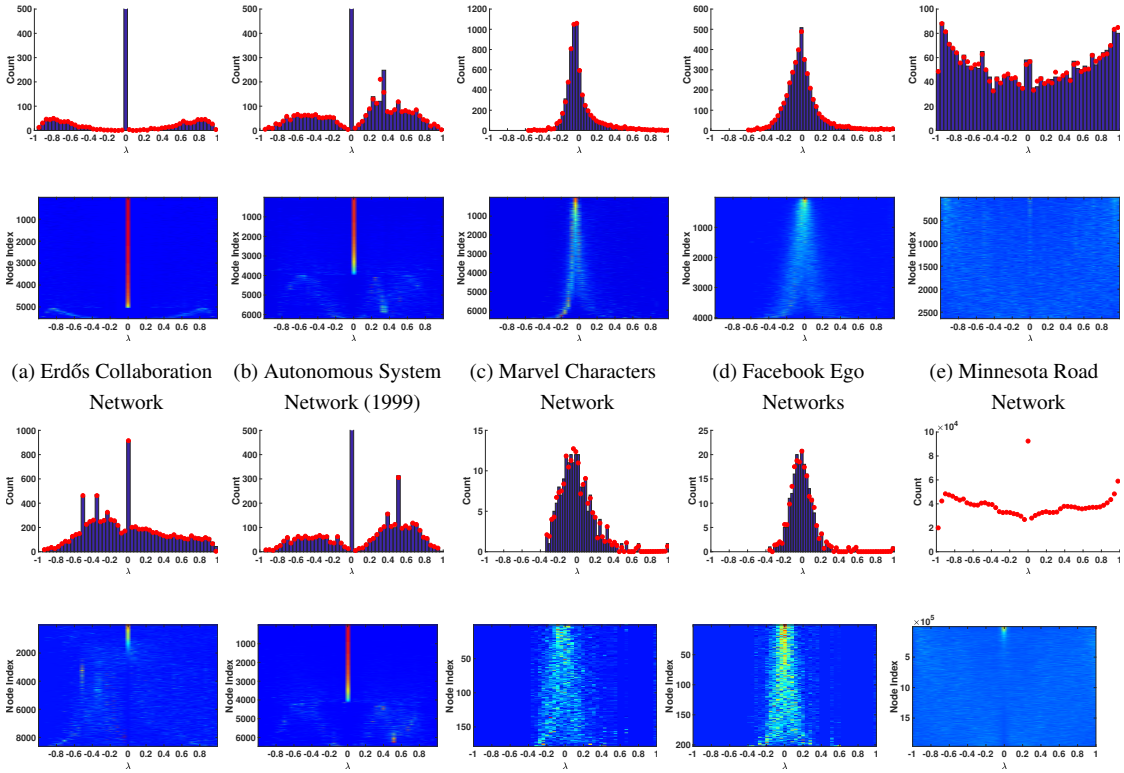
3.5 Experiments

3.5.1 Gallery of DOS/PDOS

We first present our spectral histogram approximation from DOS/ PDOS on a wide variety of graphs, including collaboration networks, online social networks, road networks and autonomous systems (dataset details are in the appendix). For all examples, we apply our methods to the normalized adjacency matrices using 500 Chebyshev moments and 20 Hadamard probe vectors. Afterwards, the spectral density is integrated into 50 histogram bins. In Fig. 3.4, the DOS approximation is on the first row, and the PDOS approximation is on the second. When a spike exists in the spectrum, we apply motif filtering, and DOS is zoomed appropriately to show the remaining part. For PDOS, we stack the spectral histograms for all nodes vertically, sorted by their projected weights on the leading left singular vector. Red indicates that a node has high weight at certain parts of the spectrum, whereas blue indicates low weight.

We observe many distinct shapes of spectrum in our examples. The eigenvalues of denser graphs, such as the Marvel characters network (3.4c: average degree 52.16) and Facebook union of ego networks (3.4d: average degree 43.69), exhibit decay similar to thepower-law around $\lambda = 0$. There has been study on the power-law distribution in the eigenvalues of the adjacency and the Laplacian matrix, but it only focuses on the leading eigenvalues rather than the entire spectrum [57] for large real-world datasets. Relatively sparse graphs (3.4a: average degree 3.06,; 3.4b: average degree 4.13) often possess spikes, especially around $\lambda = 0$, which reflect a larger set of loosely-connected boundary nodes. It is much more evident in the PDOS spectral histograms, which allow us to pick out the nodes with dominant weights at $\lambda = 0$ and those that contribute most to local structures. Finally, though the road network is quite sparse (ave. deg 2.50), its

regularity results in a lack of special features, and most nodes contribute evenly to the spectrum according to PDOS.



(a) Erdős Collaboration Network (b) Autonomous System Network (1999) (c) Marvel Characters Network (d) Facebook Ego Networks (e) Minnesota Road Network

(f) HepTh Collaboration Network (g) Autonomous System Network (2000) (h) Harry Potter Characters Network (i) Twitter Ego Networks (j) California Road Network

Figure 3.4: DOS(top)/PDOS(bottom) histograms for the normalized adjacency of 10 networks from five domains. For DOS, blue bars are the true spectrum, and red points are from KPM (500 moments and 20 Hadamard probes). For PDOS, the spectral histograms of all nodes are aligned vertically. Red indicates high weight around an eigenvalue, and blue indicates low weight. The true spectrum for the California Road Network (3.4j) is omitted, as it is too large to compute exactly (1,965,206 nodes).

3.5.2 Computation Time

In this experiment, we show the scaling of our methods by applying them to graphs of varying size of nodes, edges, and sparsity patterns. Rather than computation power, the memory cost of loading a graph with 100M-1B edges is more often the constraint.

Hence, we report runtimes for a Python version on a Google Cloud instance with 200GB memory and an Intel Xeon E5 v3 CPU at 2.30GHz.

The datasets we use are obtained from the SNAP repository [113]. For each graph, we compute the first 10 Chebyshev moments using KPM with 20 probe vectors. Most importantly, the cost for each moment is independent of the total number of moments we compute. Table 3.1 reports number of nodes, number of edges, average degree of nodes, and the average runtime for computing each moment. We can observe that the runtime is in accordance with the theoretical complexity $O(N_z(|V| + |E|))$. For the Friendster social network with about 1.8 billion edges, computing each moment takes about 1000 seconds to compute, which means we could obtain a rough approximation to its spectrum within a day. As the dominant cost is matrix-matrix multiplication and we use several probe vectors, our approach has ample opportunity for parallel computation.

Table 3.1: Average Computation Time per Chebyshev Moment for Graphs from the SNAP Repository[α].

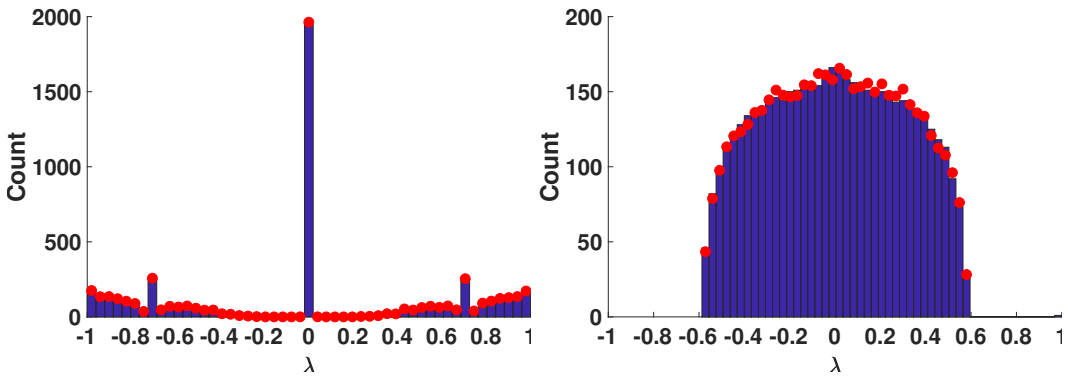
Network	# Nodes	# Edges	Avg. Deg.	Time (s)
Facebook	4,039	88,234	43.69	0.007
AstroPh	18,772	198,110	21.11	0.028
Enron	36,692	183,831	10.02	0.046
Gplus	107,614	13,673,453	254.12	1.133
Amazon	334,863	925,872	5.53	0.628
Neuron	1,018,524	24,735,503	48.57	9.138
RoadNetCA	1,965,206	2,766,607	2.82	2.276
Orkut	3,072,441	117,185,083	76.28	153.7
LiveJournal	3,997,962	34,681,189	17.35	14.52
Friendster	65,608,366	1,806,067,135	55.06	1,017

[α] 20 probe vectors are used throughout the experiment. The runtime is averaged over 5 moments.

3.5.3 Model Verification

In this experiment, we investigate the spectrum for some of the popular graph models, and whether they resemble the behavior of real-world data. Two of the most popular models used to describe real-world graphs are the scale-free model [18] and the small-world model [180]. Farkas et al. [61] has analyzed the spectrum of the adjacency matrix; we instead consider the normalized adjacency.

The scale-free model grows a random graph with the preferential attachment process, starting from an initial seed graph and adding one node and m edges at every step. Fig. 3.5 shows spectral histograms for this model with 5000 nodes and different choices of m. When $m = 1$, the generated graph has abundant local motifs like many sparse real-world graphs. By searching in PDOS for the nodes that have high weight at the two spikes, we find node-doubles ($\lambda = 0$) and singly-attached chains ($\lambda = \pm 1/\sqrt{2}$). When $m = 5$, the graph is denser, without any particular motifs, resulting in an approximately semicircular spectral distribution.

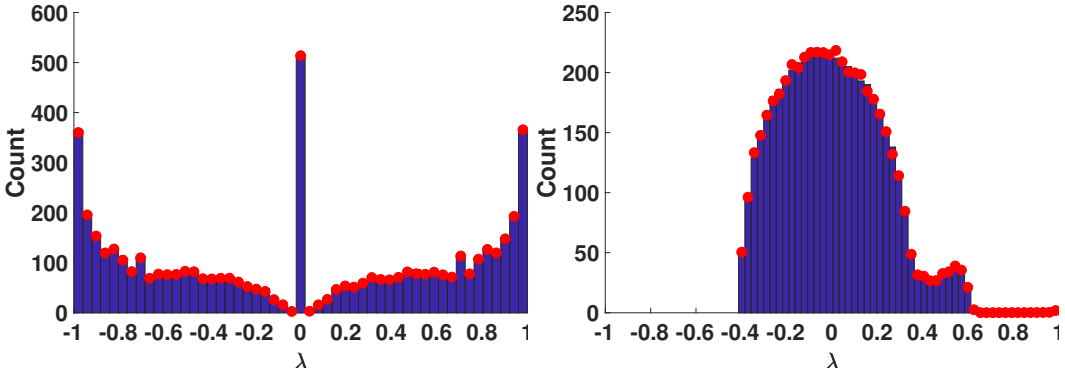


(a) $m = 1$ (b) $m = 5$

Figure 3.5: Spectral histogram for scale-free model with 5000 nodes and different m. Blue bars are the real spectrum, red points are from KPM (500 moments and 20 probes).

The small-world model generates a random graph by re-wiring edges of a ring lattice with a certain probability p. Here we construct these graphs on 5000 nodes with $p =$

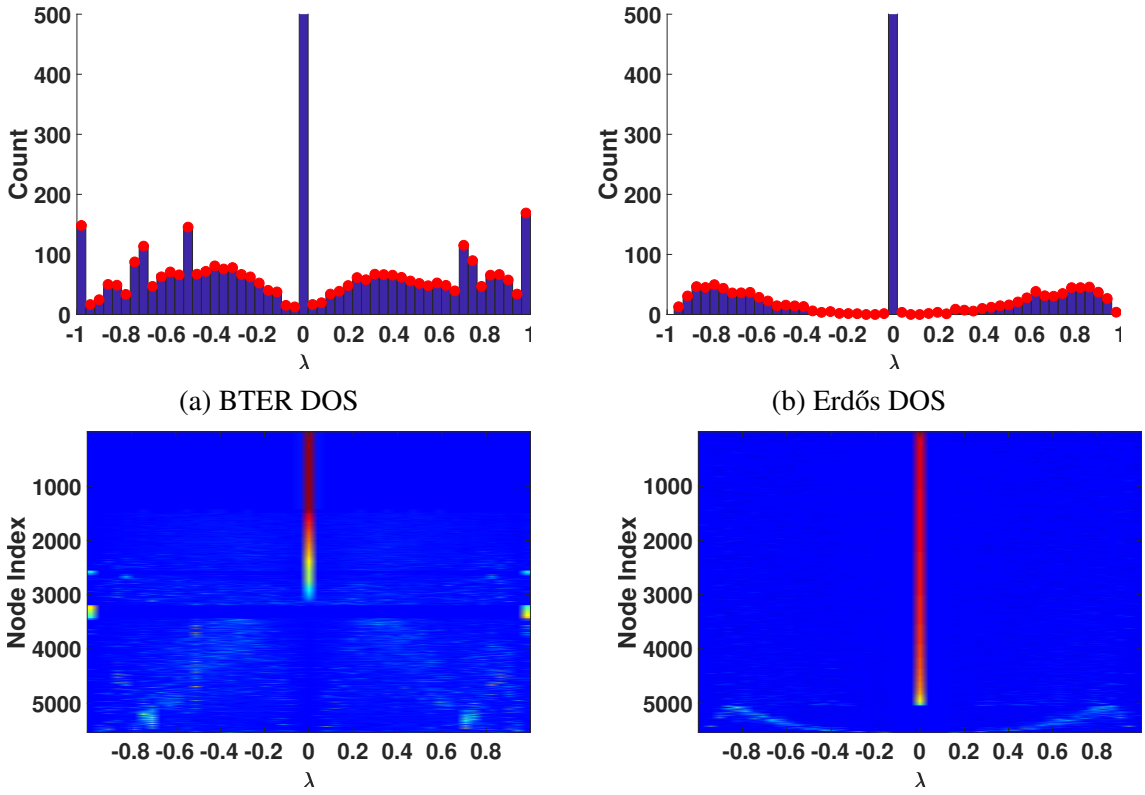
0.5; the pattern in spectrum is insensitive for a wide range of p. In Fig. 3.6, when the graph is sparse with 5000 edges, the spectrum has spikes at 0 and ± 1, indicating local symmetries, bipartite structure, and disconnected components. With 50000 edges, localized structures disappear and the spectrum has narrower support.



(a) $|E| = 5k$ (b) $|E| = 50k$

Figure 3.6: Spectral histograms for small-world model with 5000 nodes and re-wiring probability $p = 0.5$, starting with 5000 (3.6a) and 50000 (3.6b edges. Blue bars are the real spectrum, red points are from KPM (5000 moments and 20 probes).

Finally, we investigate the Block Two-Level Erdős-Rényi (BTER) model [165], which directly fits an input graph. BTER constructs a similar graph by a two-step process: first create a collection of Erdős-Rényi subgraphs, then interconnect those using a Chung-Lu model [36]. Seshadhri et al. showed their model accurately captures the observable properties of the given graph, including the eigenvalues of the adjacency matrix. Fig. 3.7 compares the DOS/PDOS of the Erdős collaboration network and its BTER counterpart. Unlike the original graph, most 0 eigenvalues in BTER graph come from isolated nodes. The BTER graph also has many more isolated edges ($\lambda = \pm 1$), singly-attached chains ($\lambda = \pm 1/\sqrt{2}$), and singly-attached triangles ($\lambda = -1/2$). We locate these motifs by inspecting nodes with high weights at respective part of the spectrum.



(a) BTER DOS (b) Erdős DOS

(c) BTER PDOS (d) Erdős PDOS

Figure 3.7: Comparison of spectral histogram between Erdős Collaboration Network and the BTER model. Both DOS and PDOS are computed with 500 moments and 20 probe vectors.

3.6 Conclusion

In this chapter, we make the computation of spectral densities a practical tool for the analysis of large real-world network. Our approach borrows from methods in solid state physics, but with adaptations that improve performance in the network analysis setting by special handling of graph motifs that leave distinctive spectral fingerprints. We show that the spectral densities are stable to small changes in the graph, as well as providing an analysis of the approximation error in our methods. We illustrate the efficiency of our approach by treating graphs with tens of millions of nodes and billions of edges using only a single compute node. The method provides a compelling visual fingerprint of a

graph, and we show how this fingerprint can be used for tasks such as model verification.

Our approach opens the door for the use of complete spectral information in large-scale network analysis. It provides a framework for scalable computation of quantities already used in network science, such as common centrality measures and graph connectivity indices (such as the Estrada index) that can be expressed in terms of the diagonals and traces of matrix functions. But we expect it to serve more generally to define new families of features that describe graphs and the roles nodes play within those graphs. We have shown that graphs from different backgrounds demonstrate distinct spectral characteristics, and thus can be clustered based on those. Looking at LDOS across nodes for role discovery, we can identify the ones with high similarity in their local structures. Moreover, extracting nodes with large weights at various points of the spectrum uncovers motifs and symmetries. In the future, we expect to use DOS/LDOS as graph features for applications in graph clustering, graph matching, role classification, and other tasks.

Acknowledgments. We thank NSF DMS-1620038 for supporting this work.

CHAPTER 4

SCALABLE GAUSSIAN PROCESSES

LIST OF SYMBOLS

D	Diagonal correction	Page 55
H	Preconditioner	Page 62
K	Kernel matrix	Page 47
K^{∇}	Kernel matrix with derivative information	Page 49
M	Number of inducing points	Page 46
N	Number of data points	Page 43
P	Projection matrix	Page 63
U	Interpolation points	Page 50
W	Interpolation weight matrix	Page 51
$\tilde{K}$	Kernel matrix with noise	Page 49
$\mathcal{L}$	Log marginal likelihood	Page 49
ℓ	Length scale	Page 47
μ	Mean function	Page 47
μ^{∇}	Mean function with derivative information	Page 49
σ	Noise variance	Page 49
θ	Hyper-parameters	Page 49
c_j	Chebyshev moments	Page 52
d	Dimension of data points	Page 43
f	Object function	Page 47
k	Covariance function	Page 47
k^{∇}	Covariance function with derivative information	Page 49
s_f	Signal variance	Page 47
v	Predictive variance	Page 85
x	Data points	Page 47
y	Observed function value	Page 49
z	Probe vector	Page 52

For applications as varied as Bayesian neural networks, determinantal point processes, elliptical graphical models, and kernel learning for Gaussian processes, one must compute a log determinant of an $N \times N$ positive definite matrix, and its derivatives – leading to prohibitive $O(N^3)$ computations. We propose novel $O(N)$ approaches to estimating these quantities from only fast matrix-vector multiplications. These stochastic approximations are based on Chebyshev, Lanczos, and surrogate models, and converge quickly even for kernel matrices that have challenging spectra. We leverage these approximations to develop a scalable Gaussian process approach to kernel learning. We find that Lanczos is generally superior to Chebyshev for kernel learning, and that a surrogate approach can be highly efficient and accurate with popular kernels.

On the other hand, gradient information greatly enhances the performance of Gaussian processes in many applications, e.g., Bayesian optimization, implicit surface reconstruction, and terrain reconstruction. Fitting a Gaussian processes to function values and derivatives at N points in d dimensions requires linear solves and log determinants with an $N(d + 1) \times N(d + 1)$ positive definite matrix. Hence, the complexity for direct methods is $O(N^3 d^3)$, scaling not only with the number of data points but also the number of dimensions. We adapt our methods with fast $O(Nd)$ matrix-vector multiplications, together with pivoted Cholesky preconditioning that cuts the iterations to convergence by several orders of magnitude, allowing for fast kernel learning and prediction. Our approaches, together with dimensionality reduction, allows us to scale Bayesian optimization with derivatives to high-dimensional problems and large evaluation budgets.

4.1 Introduction

There is a pressing need for scalable machine learning approaches to extract rich statistical structure from large datasets. A common bottleneck — arising in determinantal point processes [106], Bayesian neural networks [125], model comparison [126], graphical models [162], and Gaussian process kernel learning [156] — is computing a log determinant over a large positive definite matrix. While we can approximate log determinants by existing stochastic expansions relying on matrix-vector multiplications (MVMs), these approaches make assumptions, such as near-uniform eigenspectra [29], which are unsuitable in machine learning contexts. For example, the popular RBF kernel gives rise to rapidly decaying eigenvalues. Moreover, while standard approaches, such as stochastic power series, have reasonable asymptotic complexity in the rank of the matrix, they require too many terms (MVMs) for the precision necessary in machine learning applications.

Gaussian processes (GPs) provide a principled probabilistic kernel learning framework, for which a log determinant is of foundational importance. Specifically, the *marginal likelihood* of a Gaussian process is the probability of data given only kernel hyper-parameters. This utility function for kernel learning compartmentalizes into automatically calibrated model fit and complexity terms — called *automatic Occam's razor* — such that the simplest models which explain the data are automatically favored [157, 156], without the need for approaches such as cross-validation, or regularization, which can be costly, heuristic, and involve substantial hand-tuning and human intervention. The automatic complexity penalty, called the *Occam's factor* [126], is a log determinant of a kernel (covariance) matrix, related to the volume of solutions that can be expressed by the Gaussian process. Unfortunately, calculating log determinant is usually very expensive for huge matrices. The exact kernel learning costs of $O(N^3)$ flops

and the prediction cost of $O(N)$ flops per test point are clearly computationally infeasible for large datasets. As a result, GPs have traditionally been limited to a few thousand data points.

In addition, derivative information greatly enhances the performance of GPs in many applications, including Bayesian Optimization (BO) [192], implicit surface reconstruction [123], and terrain reconstruction. For many simulation models, derivatives may be computed at little extra cost via finite differences, complex step approximation, an adjoint method, or algorithmic differentiation [65]. Hence, there are ample opportunities in learning better GP models through exploiting derivative information in practice. However, the computation becomes more challenging if we consider GPs with both function value and derivative information, in which case training and prediction become $O(N^3d^3)$ and $O(Nd)$ respectively for data points in d-dimensional space [156, §9.4].

Many current approaches to scalable Gaussian processes [e.g., 154, 109, 83] focus on inference assuming a fixed kernel, or use approximations that do not allow for very flexible kernel learning [188], due to poor scaling with number of basis functions or inducing points. Alternatively, approaches which exploit algebraic structure in kernel matrices can provide highly expressive kernel learning [190], but are essentially limited to grid structured data. On the other hand, while many scalable approximation methods for Gaussian process regression have been proposed, scalable methods incorporating derivatives have received little attention.

Recently, Wilson and Nickisch [189] proposed the *structured kernel interpolation* (SKI) framework, which generalizes structuring exploiting methods to arbitrarily located data. SKI works by providing accurate and fast matrix-vector multiplies (MVMs) with kernel matrices, which can then be used in iterative solvers such as linear conjugate gradients for scalable GP inference. However, evaluating the marginal likelihood and its

derivatives, for kernel learning, has followed a scaled eigenvalue approach [190, 189] instead of iterative MVM approaches. This approach can be inaccurate, and relies on a fast eigendecomposition of a structured matrix, which is not available in many consequential situations where fast MVMs are available, including: (i) additive covariance functions, (ii) multi-task learning, (iii) change-points [84], and (iv) diagonal corrections to kernel approximations [168]. Fiedler [63] and Weyl [185] bounds have been used to extend the scaled eigenvalue approach [64, 84], but are similarly limited. These extensions are often very approximate, and do not apply beyond sums of two and three matrices, where each matrix in the sum must have a fast eigendecomposition.

In machine learning there has recently been renewed interest in MVM based approaches to approximating log determinants, such as the Chebyshev [80] and Lanczos[175] based methods, although these approaches go back at least two decades in quantum chemistry computations [14]. Independently, several authors have proposed various methods to compute derivatives of log determinants [124, 169]. But *both* the log determinant *and* the derivatives are needed for efficient GP marginal likelihood learning: the derivatives are required for gradient-based optimization, while the log determinant itself is needed for model comparison, comparisons between the likelihoods at local maximizers, and fast and effective choices of starting points and step sizes in a gradient-based optimization algorithm.

In this chapter, we develop novel scalable and general purpose Chebyshev, Lanczos, and surrogate approaches for efficiently and accurately computing both the log determinant and its derivatives simultaneously. Our methods use only fast MVMs, and re-use the same MVMs for both computations. We also propose scalable methods for GPs with derivative information built on the the SKI framework. As the uniform grids in SKI scale poorly to high-dimensional spaces, we further extend the structured kernel interpolation

for products (SKIP) method, which approximates a high-dimensional product kernel as a Hadamard product of low rank Lanczos decompositions [66]. Both SKI and SKIP provide fast approximate kernel MVMs, which are a building block to solve linear systems with the kernel matrix and to approximate log determinants [51]. In particular, our contributions are:

- We derive fast methods for simultaneously computing the log determinant and its derivatives by stochastic Chebyshev, stochastic Lanczos, and surrogate models, from MVMs alone. We also perform an error analysis and extend these approaches to higher order derivatives.
- These methods enable fast GP kernel learning whenever fast MVMs are possible, including applications where alternatives such as scaled eigenvalue methods (which rely on fast eigendecompositions) are not, such as for (i) diagonal corrections for better kernel approximations, (ii) additive covariances, (iii) multi-task approaches, and (iv) non-Gaussian likelihoods.
- We extend SKI to incorporate derivative information, enabling $O(Nd)$ complexity learning and $O(1)$ prediction per test points, relying only on fast MVM with the kernel matrix.
- We also extend SKIP, which enables scalable Gaussian process regression with derivatives in high-dimensional spaces without grids. Our approach allows for $O(Nd)$ MVMs after the inclusion of derivative information.
- We illustrate that preconditioning is critical for fast convergence of iterations for kernel matrices with derivatives. A pivoted Cholesky preconditioner cuts the iterations to convergence by several orders of magnitude when applied to both SKI and SKIP with derivatives.

- For GPs without derivative information, we illustrate the performance of our approach on several large, multi-dimensional datasets, including a consequential crime prediction problem, and a precipitation problem with $N = 528,474$ training points. We consider a variety of kernels, including deep kernels [191], diagonal corrections, and both Gaussian and non-Gaussian likelihoods.
- For GPs with derivative information, we illustrate the scalability of our approach on several examples including implicit surface fitting of the Stanford bunny, rough terrain reconstruction, and Bayesian optimization. We show how our methods, together with active subspace techniques, can be used to extend Bayesian optimization to high-dimensional problems with large evaluation budgets.
- We have released code and tutorials as an extension to the GPML library [158] at `https://github.com/kd383/GPML_SLD`. A Python implementation of our approach is also available through the GPyTorch library: `https://github.com/jrg365/gpytorch`. The code for GPs with derivative information is available at `https://github.com/ericlee0803/GP_Derivatives`.

When using our approach in conjunction with SKI [189] for fast MVMs, derivative-free GP kernel learning is $O(N + g(M))$, for M inducing points and N training points, where $g(M) \leq M \log M$. With algebraic approaches such as SKI we also do not need to worry about quadratic storage in inducing points, since symmetric Toeplitz and Kronecker matrices can be stored with at most linear cost, without needing to explicitly construct a matrix.

Although we here use SKI for fast MVMs, we emphasize that the proposed iterative approaches are generally applicable, and can easily be used in conjunction with *any* method that admits fast MVMs, including classical inducing point methods [154], finite basis expansions [109], and the popular stochastic variational approaches [83]. More-

over, stochastic variational approaches can naturally be combined with SKI to further accelerate MVMs [187].

We start in Section 4.2 with an introduction to GPs and kernel approximations. In Section 4.3 we introduce stochastic trace estimation, Chebyshev (Section 4.3.1) and Lanczos (Section 4.3.2) approximations, and various techniques we use to efficiently evaluate log determinant for kernel matrices. In Section 4.4, we describe the different sources of error in our approximations. In Section 4.5, we extend the methods in Section 4.3 for GPs with derivative information. In Section 4.6 we consider experiments for derivative-free GPs on several large real-world data sets. In Section 4.7, we demonstrate the benefit of including derivative information, as well as the performance of our methods in this case. We conclude in Section 4.8.

4.2 Background

A Gaussian process (GP) is a collection of random variables, any finite number of which have a joint Gaussian distribution [e.g., 156]. A GP can be used to define a distribution over functions $f(x) \sim \mathcal{GP}(\mu(x), k(x, x'))$, where each function value is a random variable indexed by $x \in \mathbb{R}^d$, and $\mu : \mathbb{R}^d \to \mathbb{R}$ and $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ are the mean and covariance functions of the process.

Two popular covariance kernels are the RBF kernel

$$k_{\text{RBF}}(x, x') = s_f^2 \exp\left(\frac{\left\|x - x'\right\|^2}{2\ell^2}\right) \tag{4.1}$$

and the Matérn kernel

$$k_{\text{Mat},\nu}(x, x') = s_f^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu}\frac{\left\|x - x'\right\|}{\ell}\right)^{\nu} K_\nu \left(\sqrt{2\nu}\frac{\left\|x - x'\right\|}{\ell}\right) \tag{4.2}$$

where $1/2$, $3/2$, and $5/2$ are popular choices for ν to model heavy-tailed correlations between function values. The spectral behavior of these and other kernels has been well-studied for years, and we recommend [179] for recent results. Particularly relevant to our discussion is a theorem due to Weyl, which says that if a symmetric kernel has ν continuous derivatives, then the eigenvalues of the associated integral operator decay like $\left|\lambda_n\right| = o(n^{-\nu-1/2})$. Hence, the eigenvalues of kernel matrices for the smooth RBF kernel (and of any given covariance matrix based on that kernel) tend to decay much more rapidly than those of the less smooth Matérn kernel, which has two derivatives at zero for $\nu = 5/2$, one derivative at zero for $\nu = 3/2$, and no derivatives at zero for $\nu = 1/2$. This matters to the relative performance of Chebyshev and Lanczos approximations of the log determinant for large values of s_f and small values of σ on the exact and approximate RBF kernel. We also introduce the spline kernel that we used in one of the experiments.

$$k_{\text{spline}}(x, y) = \begin{cases} s^2(\|x-y\|^3 + a\|x-y\|^2 + b) & d \text{ odd} \\ s^2(\|x-y\|^2 \log\|x-y\| + a\|x-y\|^2 + b) & d \text{ even} \end{cases} \tag{4.3}$$

where a, b are chosen to make the spline kernel symmetric and positive definite on the given domain. We denote any kernel hyper-parameters by the vector θ. To be concise, we try to avoid explicitly denote the dependence of k and associated matrices on θ.

For any locations $X = \{x_1, \ldots, x_N\} \subset \mathbb{R}^d$, $f_X \sim \mathcal{N}(\mu_X, K_{XX})$ where f_X and μ_X represent the vectors of function values for f and μ evaluated at each of the $x_i \in X$, and K_{XX} is the matrix whose (i, j) entry is $k(x_i, x_j)$. Suppose we have a vector of corresponding function values $y \in \mathbb{R}^N$, where each entry is contaminated by independent Gaussian noise with variance σ^2. Under a Gaussian process prior depending on the covariance hyper-parameters θ, the log marginal likelihood is given by

$$\mathcal{L}(\theta|y) = -\frac{1}{2}\left[(y-\mu_X)^T\alpha + \log\left|\tilde{K}_{XX}\right| + N\log 2\pi\right], \tag{4.4}$$

where $\alpha = \tilde{K}_{XX}^{-1}(y - \mu_X)$ and $\tilde{K}_{XX} = K_{XX} + \sigma^2 I$. Optimization of (4.4) is expensive, since the cheapest way of evaluating $\log\left|\tilde{K}_{XX}\right|$ and its derivatives without taking advantage of the structure of $\tilde{K}_{XX}$ involves computing the $O(N^3)$ Cholesky factorization of $\tilde{K}_{XX}$. $O(N^3)$ computations is too expensive for inference and learning beyond even just a few thousand points.

Meanwhile, differentiation is a linear operator, and (assuming a twice-differentiable kernel) we may define a multi-output GP for the function and (scaled) gradient values with mean and kernel functions

$$\mu^{\nabla}(x) = \begin{bmatrix} \mu(x) \\ \partial_x \mu(x) \end{bmatrix}, \qquad k^{\nabla}(x, x') = \begin{bmatrix} k(x, x') & (\partial_{x'} k(x, x'))^T \\ \partial_x k(x, x') & \partial^2 k(x, x') \end{bmatrix}, \tag{4.5}$$

where $\partial_x k(x, x')$ and $\partial^2 k(x, x')$ represent the column vector of (scaled) partial derivatives in x and the matrix of (scaled) second partials in x and x', respectively. Scaling derivatives by a natural length scale gives the multi-output GP consistent units, and lets us understand approximation error without weighted norms. As in the scalar GP case, we model measurements of the function as contaminated by independent Gaussian noise.

Because the kernel matrix for the GP on function values alone is a submatrix of the kernel matrix for function values and derivatives together, the predictive variance in the presence of derivative information will be strictly less than the predictive variance without derivatives. Hence, convergence of regression with derivatives is always superior to convergence of regression without, which is well-studied in, e.g. [156, Chapter 7]. Fig. 4.1 illustrates the value of derivative information; fitting with derivatives is evidently much more accurate than fitting function values alone. In higher-dimensional problems, derivative information is even more valuable, but it comes at a cost: the kernel matrix K_{XX}^{∇} is of size $N(d + 1)$-by-$N(d + 1)$. Scalable approximate solvers are therefore vital in order to use GPs for large datasets with derivative data, particularly in high-dimensional spaces.

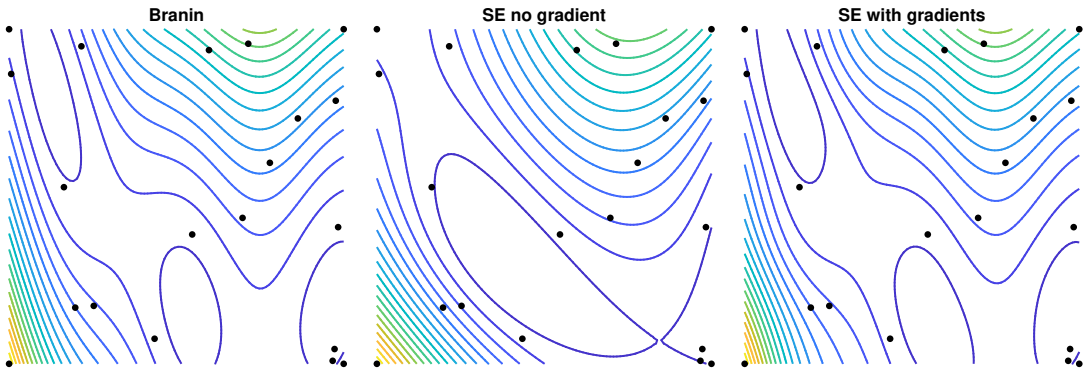


Figure 4.1: An example where gradient information pays off; the true function is on the left. Compare the regular GP without derivatives (middle) to the GP with derivatives (right). Unlike the former, the latter is able to accurately capture critical points of the function.

A popular approach to GP scalability is to replace the exact kernel $k(x, z)$ by an approximate kernel that admits fast computations [154]. Several methods approximate $k(x, z)$ via *inducing points* $U = \{u_j\}_{j=1}^{M} \subset \mathbb{R}^d$. An example is the subset of regressor (SoR) kernel:

$$k^{\text{SoR}}(x, z) = K_{xU} K_{UU}^{-1} K_{Uz}\,, \tag{4.6}$$

which is a low-rank approximation [166]. The SoR matrix $\tilde{K}_{XX}^{\text{SoR}} \in \mathbb{R}^{N \times N}$ has rank at most M, allowing us to solve linear systems involving $\tilde{K}_{XX}^{\text{SoR}} = K_{XX}^{\text{SoR}} + \sigma^2 I$ and to compute $\log \left|\tilde{K}_{XX}^{\text{SoR}}\right|$ in $O(M^2 N + M^3)$ time. Another popular kernel approximation is the fully independent training conditional (FITC), which is a diagonal correction of SoR so that the diagonal is the same as for the original kernel [168]. Thus kernel matrices from FITC have low-rank plus diagonal structure. This modification has had exceptional practical significance, leading to improved point predictions and much more realistic predictive uncertainty [154, 155], making FITC arguably the most popular approach for scalable Gaussian processes.

Wilson and Nickisch [189] provides a mechanism for fast MVMs through proposing

the structured kernel interpolation (SKI) approximation,

$$K_{XX} \approx W K_{UU} W^T , \tag{4.7}$$

where W is an N-by-M matrix of interpolation weights; the authors of [189] use local cubic interpolation so that W is sparse. The sparsity in W makes it possible to naturally exploit algebraic structure (such as Kronecker or Toeplitz structure) in K_{UU} when the inducing points U are on a grid, for extremely fast matrix vector multiplications with the approximate K_{XX} even if the data inputs X are arbitrarily located. For instance, if K_{UU} is Toeplitz, then each MVM with the approximate K_{XX} costs only $O(N + M \log M)$. By contrast, placing the inducing points U on a grid for classical inducing point methods, such as SoR or FITC, does not result in substantial performance gains, due to the costly cross-covariance matrices K_{xU} and K_{Uz}. A limitation of SKI is that the number of grid points increases exponentially with the dimension. This exponential scaling has been addressed by structured kernel interpolation for products (SKIP) [66], which decomposes the kernel matrix for a product kernel in d-dimensions as a Hadamard (elemen-wise) product of one-dimensional kernel matrices.

4.3 Methods: Derivative-Free GPs

Our goal is to estimate, for a symmetric positive definite matrix $\tilde{K}$,

$$\log\left|\tilde{K}\right| = \mathrm{tr}(\log(\tilde{K})) \quad \text{and} \quad \frac{\partial}{\partial\theta_i}\left[\log\left|\tilde{K}\right|\right] = \mathrm{tr}\left(\tilde{K}^{-1}\left(\frac{\partial\tilde{K}}{\partial\theta_i}\right)\right), \tag{4.8}$$

where log is the matrix logarithm [86]. We compute the traces involved in both the log determinant and its derivative via *stochastic trace estimators* [96], which approximate the trace of a matrix using only matrix-vector products.

The key idea is introduced in Section 2.3: for a given matrix A and a random probe

vector z with independent entries with mean zero and variance one, then $\mathrm{tr}(A) = \mathbb{E}[z^T Az]$. In this section, we let the entries of the probe vectors be Rademacher random variables. We also estimate the trace by the sample mean over N_z independent probe vectors. Often surprisingly few probe vectors suffice.

To estimate $\mathrm{tr}(\log(\tilde{K}))$, we need to multiply $\log(\tilde{K})$ by probe vectors. We consider two ways to estimate $\log(\tilde{K})z$: by a polynomial approximation of log or by using the connection between the Gaussian quadrature rule and the Lanczos method [80, 175] (Section 2.2). In both cases, we show how to re-use the same probe vectors for an inexpensive coupled estimator of the derivatives. In addition, we may use standard radial basis function interpolation of the log determinant evaluated at a few systematically chosen points in the hyper-parameter space as an inexpensive surrogate for the log determinant.

4.3.1 Chebyshev Approximation

As given in Eq. (2.2), Chebyshev polynomials are defined by the recursion

$$T_0(x) = 1, \quad T_1(x) = x, \quad T_{j+1}(x) = 2xT_j(x) - T_{j-1}(x) \;\text{ for }\; j \geq 1\,. \tag{4.9}$$

For $f : [-1, 1] \to \mathbb{R}$ the Chebyshev interpolant of degree m is

$$f(x) \approx p_m(x) := \sum_{j=0}^{m} c_j T_j(x)\,, \quad \text{where } c_j = \frac{2 - \delta_{j0}}{m+1} \sum_{k=0}^{m} f(x_k) T_j(x_k)\,, \tag{4.10}$$

and δ_{j0} is the Kronecker delta and $x_k = \cos(\pi(k + 1/2)/(m + 1))$ for $k = 0, 1, 2, \ldots, m$; see [68]. Using the Chebyshev interpolant of $\log(1 + \alpha x)$, we approximate $\log\left|\tilde{K}\right|$ by

$$\log\left|\tilde{K}\right| - n\log\beta = \log|I + \alpha B| \approx \sum_{j=0}^{m} c_j\,\mathrm{tr}(T_j(B))\,, \tag{4.11}$$

when $B = (\tilde{K}/\beta - 1)/\alpha$ has eigenvalues $\lambda_i \in (-1, 1)$.

For stochastic estimation of $\mathrm{tr}(T_j(B))$, we only need to compute $z^T T_j(B) z$ for each given probe vector z. We compute vectors $w_j = T_j(B)z$ and $\partial w_j / \partial \theta_i$ via the coupled recurrences

$$w_0 = z\,, \qquad w_1 = Bz\,, \qquad w_{j+1} = 2Bw_j - w_{j-1} \text{ for } j \geq 1\,, \tag{4.12}$$

$$\frac{\partial w_0}{\partial \theta_i} = 0\,, \quad \frac{\partial w_1}{\partial \theta_i} = \frac{\partial B}{\partial \theta_i} z\,, \quad \frac{\partial w_{j+1}}{\partial \theta_i} = 2\left(\frac{\partial B}{\partial \theta_i} w_j + B\frac{\partial w_j}{\partial \theta_i}\right) - \frac{\partial w_{j-1}}{\partial \theta_i} \text{ for } j \geq 1\,. \tag{4.13}$$

This gives the estimators

$$\log\left|\tilde{K}\right| \approx \mathbb{E}\left[\sum_{j=0}^{m} c_j z^T w_j\right] \quad \text{and} \quad \frac{\partial}{\partial \theta_i} \log\left|\tilde{K}\right| \approx \mathbb{E}\left[\sum_{j=0}^{m} c_j z^T \frac{\partial w_j}{\partial \theta_i}\right]. \tag{4.14}$$

Thus, each derivative of the approximation costs two extra MVMs per term.

4.3.2 Gauss Quadrature and Lanczos

We can also approximate $z^T \log(\tilde{K}) z$ via a Lanczos decomposition; see [73] for discussion of a Lanczos-based computation of $z^T f(\tilde{K}) z$ and [175, 14] for stochastic Lanczos estimation of log determinants. We run m steps of the Lanczos algorithm, which computes the decomposition

$$\tilde{K} Q_m = Q_m \Gamma_m + \beta_m q_{m+1} e_m^T\,, \tag{4.15}$$

where $Q_m = [q_1, q_2, \cdots, q_m] \in \mathbb{R}^{n \times m}$ is a matrix with orthonormal columns such that $q_1 = z/\left\|z\right\|$, $\Gamma_m \in \mathbb{R}^{m \times m}$ is tridiagonal, β_m is the residual, and e_m is the m-th Cartesian unit vector. We estimate

$$z^T f(\tilde{K}) z \approx e_1^T f(\left\|z\right\|^2 \Gamma_m) e_1\,, \tag{4.16}$$

where e_1 is the first column of the identity. Because the Lanczos algorithm is numerically unstable, there exist several practical implementations to resolve this issue [42, 163]. The approximation (4.16) corresponds to a Gauss quadrature rule for the Riemann-Stieltjes integral of the measure associated with the eigenvalue distribution of $\tilde{K}$.

It is exact when f is a polynomial of degree up to $2m-1$. This approximation is also exact when $\tilde{K}$ has at most m distinct eigenvalues, which is particularly relevant to Gaussian process regression, since frequently the kernel matrices only have a small number of eigenvalues that are not close to zero.

The Lanczos decomposition also allows us to estimate derivatives of the log determinant at minimal cost. Via the Lanczos decomposition, we have

$$\hat{g} = Q_m(\Gamma_m^{-1} e_1 \|z\|) \approx \tilde{K}^{-1} z\,. \tag{4.17}$$

This approximation requires no additional matrix vector multiplications beyond those used to compute the Lanczos decomposition, which we already used to estimate $\log(\tilde{K})z$; in exact arithmetic, this is equivalent to m steps of conjugate gradient (CG). Computing $\hat{g}$ in this way takes $O(mN)$ additional time; subsequently, we only need one matrix-vector multiply by $\partial\tilde{K}/\partial\theta_i$ for each probe vector to estimate $\mathrm{tr}(\tilde{K}^{-1}(\partial\tilde{K}/\partial\theta_i)) = \mathbb{E}\left[(\tilde{K}^{-1}z)^T(\partial\tilde{K}/\partial\theta_i)z\right]$.

4.3.3 Diagonal Correction to SKI

The SKI approximation may provide a poor estimate of the diagonal entries of the original kernel matrix for kernels with limited smoothness, such as the Matérn kernel. In general, diagonal corrections to scalable kernel approximations can lead to great performance gains. Indeed, the popular FITC method [168] is exactly a diagonal correction of SoR.

We thus modify the SKI approximation to add a diagonal matrix D,

$$K_{XX} \approx W K_{UU} W^T + D\,, \tag{4.18}$$

such that the diagonal of the approximated K_{XX} is exact. In other words, D subtracts the diagonal of $WK_{UU}W^T$ and adds the true diagonal of K_{XX}. This modification is not possible for the scaled eigenvalue method for approximating log determinants in [189], since adding a diagonal matrix makes it impossible to approximate the eigenvalues of K_{XX} from the eigenvalues of K_{UU}.

However, Eq. 4.18 still admits fast MVMs and thus works with our approach for estimating the log determinant and its derivatives. Computing D with SKI costs only $O(n)$ flops since W is sparse for local cubic interpolation. We can therefore compute $(W^T e_i)^T K_{UU}(W^T e_i)$ in $O(1)$ flops.

4.3.4 Estimating Higher Derivatives

We have already described how to use stochastic estimators to compute the log marginal likelihood and its first derivatives. The same approach applies to computing higher-order derivatives for a Newton-like iteration, to understand the sensitivity of the maximum likelihood parameters, or for similar tasks. The first derivatives of the full log marginal likelihood are

$$\frac{\partial \mathcal{L}}{\partial \theta_i} = -\frac{1}{2}\left[\operatorname{tr}\left(\tilde{K}^{-1}\frac{\partial \tilde{K}}{\partial \theta_i}\right) - \alpha^T \frac{\partial \tilde{K}}{\partial \theta_i}\alpha\right], \tag{4.19}$$

and the second derivatives of the two terms are

$$\frac{\partial^2}{\partial \theta_i \partial \theta_j}\left[\log|\tilde{K}|\right] = \operatorname{tr}\left(\tilde{K}^{-1}\frac{\partial^2 \tilde{K}}{\partial \theta_i \partial \theta_j} - \tilde{K}^{-1}\frac{\partial \tilde{K}}{\partial \theta_i}\tilde{K}^{-1}\frac{\partial \tilde{K}}{\partial \theta_j}\right), \tag{4.20}$$

$$\frac{\partial^2}{\partial \theta_i \partial \theta_j}\left[(y-\mu_X)^T\alpha\right] = 2\alpha^T \frac{\partial \tilde{K}}{\partial \theta_i}\tilde{K}^{-1}\frac{\partial \tilde{K}}{\partial \theta_j}\alpha - \alpha^T \frac{\partial^2 \tilde{K}}{\partial \theta_i \partial \theta_j}\alpha\,. \tag{4.21}$$

Superficially, evaluating the second derivatives would appear to require several additional solves above and beyond those used to estimate the first derivatives of the log determinant. In fact, we can get an unbiased estimator for the second derivatives with

no additional solves, but only fast products with the derivatives of the kernel matrices. Let z and w be independent probe vectors, and define $g = \tilde{K}^{-1}z$ and $h = \tilde{K}^{-1}w$. Then

$$\frac{\partial^2}{\partial\theta_i\partial\theta_j}\left[\log\left|\tilde{K}\right|\right] = \mathbb{E}\left[g^T\frac{\partial^2\tilde{K}}{\partial\theta_i\partial\theta_j}z - \left(g^T\frac{\partial\tilde{K}}{\partial\theta_i}w\right)\left(h^T\frac{\partial\tilde{K}}{\partial\theta_j}z\right)\right], \tag{4.22}$$

$$\frac{\partial^2}{\partial\theta_i\partial\theta_j}\left[(y-\mu_X)^T\alpha\right] = 2\,\mathbb{E}\left[\left(z^T\frac{\partial\tilde{K}}{\partial\theta_i}\alpha\right)\left(g^T\frac{\partial\tilde{K}}{\partial\theta_j}\alpha\right)\right] - \alpha^T\frac{\partial^2\tilde{K}}{\partial\theta_i\partial\theta_j}\alpha\,. \tag{4.23}$$

Hence, if we use the stochastic Lanczos method to compute the log determinant and its derivatives, the additional work required to obtain a second derivative estimate is one MVM by each second partial of the kernel for each probe vector and for α, one MVM of each first partial of the kernel with α, and a few dot products.

4.3.5 Surrogate Model

Another way to deal with the log determinant and its derivatives is to evaluate the log determinant term at a few systematically chosen points in the space of hyperparameters and fit an interpolation approximation to these values. This is particularly useful when the kernel depends on a modest number of hyperparameters (e.g., half a dozen), and thus the number of points we need to precompute is relatively small. We refer to this method as a surrogate, since it provides an inexpensive substitute for the log determinant and its derivatives. For our surrogate approach, we use radial basis function (RBF) interpolation, which is one of the most popular models for approximating scattered data in a general number of dimensions [31, 62, 164, 183]. Given distinct interpolation points $\Theta = \{\theta^i\}_{i=1}^n$, the RBF model takes the form

$$s_\Theta(\theta) = \sum_{i=1}^{n}\lambda_i\varphi(\left\|x-\theta^i\right\|) + p(x)\,, \tag{4.24}$$

where the kernel $\varphi : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a one-dimensional function and $p \in \Pi_{m-1}^d$, the space of polynomials with d variables of degree no more than $m-1$. There are many possible

choices for φ, such as the cubic kernel $\varphi(r) = r^3$ that we adopt in our experiments, and the thin-plate spline kernel $\varphi(r) = r^2 \log(r)$. The coefficients λ_i are determined by imposing the interpolation conditions $s_\Theta(\theta^i) = \log\left|K(\theta^i)\right|$ for $i = 1, \ldots, n$ and the discrete orthogonality condition

$$\sum_{i=1}^{n} \lambda_i q(\theta^i) = 0, \qquad \forall q \in \Pi_{m-1}^d. \tag{4.25}$$

For appropriate RBF kernels, this linear system is nonsingular provided that polynomials in Π_{m-1}^d are uniquely determined by their values on the interpolation set. We refer the readers to this dissertation [58] for more details.

4.4 Error Properties

4.4.1 First-Order Analysis

In addition to the usual errors from sources such as solver termination criteria and floating point arithmetic, our approach to kernel learning involves several additional sources of error: we approximate the true kernel with one that enables fast MVMs, we approximate traces using stochastic estimation, and we approximate the actions of $\log(\tilde{K})$ and $\tilde{K}^{-1}$ on probe vectors.

We can compute first-order estimates of the sensitivity of the log likelihood to perturbations in the kernel using the same stochastic estimators we use for the derivatives with respect to hyper-parameters. For example, if $\mathcal{L}^{\text{ref}}$ is the likelihood for a reference kernel $\tilde{K}^{\text{ref}} = \tilde{K} + E$, then

$$\mathcal{L}^{\text{ref}}(\theta|y) = \mathcal{L}(\theta|y) - \frac{1}{2}\left(\mathbb{E}\left[g^T E z\right] - \alpha^T E \alpha\right) + O(\|E\|^2), \tag{4.26}$$

and we can bound the change in likelihood at first order by $\|E\|\left(\|g\|\,\|z\| + \|\alpha\|^2\right)$. Given bounds on the norms of $\partial E/\partial\theta_i$, we can similarly estimate changes in the gradient of the likelihood, allowing us to bound how the marginal likelihood hyperparameter estimates depend on kernel approximations.

If $\tilde{K} = U\Lambda U^T + \sigma^2 I$, the Hutchinson trace estimator has known variance [12]

$$\mathrm{Var}\left[z^T \log(\tilde{K})z\right] = \sum_{i\neq j}\left[\log(\tilde{K})\right]_{ij}^2 \leq \sum_{i=1}^{n} \log(1+\lambda_j/\sigma^2)^2\,. \tag{4.27}$$

If the eigenvalues of the kernel matrix without noise decay rapidly enough compared to σ, the variance will be small compared to the magnitude of $\mathrm{tr}(\log\tilde{K}) = 2n\log\sigma + \sum_{i=1}^{n}\log(1+\lambda_j/\sigma^2)$. Hence, we need fewer probe vectors to obtain reasonable accuracy than one would expect from bounds that are blind to the matrix structure. In our experiments, we typically only use 5–10 probes — and we use the sample variance across these probes to estimate *a posteriori* the stochastic component of the error in the log likelihood computation. If we are willing to estimate the Hessian of the log likelihood, we can increase rates of convergence for finding kernel hyper-parameters.

The Chebyshev approximation scheme requires $O(\sqrt{\kappa}\log(\kappa/\epsilon))$ steps to obtain an $O(\epsilon)$ approximation error in computing $z^T\log(\tilde{K})z$, where $\kappa = \lambda_{\max}/\lambda_{\min}$ is the condition number of $\tilde{K}$ [80]. This behavior is independent of the distribution of eigenvalues within the interval $[\lambda_{\min}, \lambda_{\max}]$, and is close to optimal when eigenvalues are spread quasi-uniformly across the interval. Nonetheless, when the condition number is large, convergence may be quite slow. The Lanczos approach converges at least twice as fast as Chebyshev in general [175, Remark 1], and converges much more rapidly when the eigenvalues are *not* uniform within the interval, as is the case with log determinants of many kernel matrices. Hence, we recommend the Lanczos approach over the Chebyshev approach in general. In all of our experiments, the error associated with approximating $z^T\log(\tilde{K})z$ by Lanczos was dominated by other sources of error.

4.4.2 Comparison to Reference Kernel

Suppose more generally that $\tilde{K} = K + \sigma^2 I$ is an approximation to a reference kernel matrix $\tilde{K}^{\text{ref}} = K^{\text{ref}} + \sigma^2 I$, and let $E = K^{\text{ref}} - K$. Let $\mathcal{L}(\theta|y)$ and $\mathcal{L}^{\text{ref}}(\theta|y)$ be the log likelihood functions for the two kernels; then

$$\mathcal{L}^{\text{ref}}(\theta|y) = \mathcal{L}(\theta|y) - \frac{1}{2}\left[\text{tr}(\tilde{K}^{-1}E) - \alpha^T E\alpha\right] + O(\|E\|^2) \tag{4.28}$$

$$\frac{\partial}{\partial\theta_i}\mathcal{L}^{\text{ref}}(\theta|y) = \frac{\partial}{\partial\theta_i}\mathcal{L}(\theta|y) - \frac{1}{2}\left[\text{tr}\left(\tilde{K}^{-1}\frac{\partial E}{\partial\theta_i} - \tilde{K}^{-1}\frac{\partial\tilde{K}}{\partial\theta_i}\tilde{K}^{-1}E\right) - \alpha^T\frac{\partial E}{\partial\theta_i}\alpha\right] + O(\|E\|^2). \tag{4.29}$$

If we are willing to pay the price of a few MVMs with E, we can use these expressions to improve our maximum likelihood estimate. Let z and w be independent probe vectors with $g = \tilde{K}^{-1}z$ and $\hat{g} = \tilde{K}^{-1}w$. To estimate the trace in the derivative computation, we use the standard stochastic trace estimation approach together with the observation that $\mathbb{E}[ww^T] = I$:

$$\text{tr}\left(\tilde{K}^{-1}\frac{\partial E}{\partial\theta_i} - \tilde{K}^{-1}\frac{\partial\tilde{K}}{\partial\theta_i}\tilde{K}^{-1}E\right) = \mathbb{E}\left[g^T\frac{\partial E}{\partial\theta_i}z - g^T\frac{\partial K}{\partial\theta_i}w\hat{g}^T Ez\right] \tag{4.30}$$

This linearization may be used directly (with a stochastic estimator); alternately, if we have an estimates for $\|E\|$ and $\left\|\partial E/\partial\theta_i\right\|$, we can substitute these in order to get estimated bounds on the magnitude of the derivatives. Coupled with a similar estimator for the Hessian of the likelihood function, we can use this method to compute the maximum likelihood parameters for the fast kernel, then compute a correction $-H^{-1}\nabla_\theta\mathcal{L}^{\text{ref}}$ to estimate the maximum likelihood parameters of the reference kernel.

4.5 Methods: GPs with Derivative Information

One standard approach to scaling GPs substitutes the exact kernel with an approximate kernel. When the GP fits values and gradients, one may attempt to separately approxi-

mate the kernel and the kernel derivatives. Unfortunately, this may lead to indefiniteness, as the resulting approximation is no longer a valid kernel. Instead, we differentiate the approximate kernel, which preserves positive definiteness. We do this for the SKI and SKIP kernels below, but our general approach applies to any approximate MVM.

4.5.1 D-SKI

D-SKI (SKI with derivatives) is the standard kernel matrix for GPs with derivatives, but applied to the SKI kernel. Equivalently, we differentiate the interpolation scheme:

$$k(x, x') \approx \sum_i w_i(x) k(x_i, x') \rightarrow \nabla k(x, x') \approx \sum_i \nabla w_i(x) k(x_i, x') \,. \tag{4.31}$$

One can use cubic convolutional interpolation [103], but higher order methods lead to greater accuracy, and we therefore use quintic interpolation [135]. The resulting D-SKI kernel matrix has the form

$$\begin{bmatrix} K & (\partial K)^T \\ \partial K & \partial^2 K \end{bmatrix} \approx \begin{bmatrix} W \\ \partial W \end{bmatrix} K_{UU} \begin{bmatrix} W \\ \partial W \end{bmatrix}^T = \begin{bmatrix} W K_{UU} W^T & W K_{UU} (\partial W)^T \\ (\partial W) K_{UU} W^T & (\partial W) K_{UU} (\partial W)^T \end{bmatrix}, \tag{4.32}$$

where the elements of sparse matrices W and ∂W are determined by $w_i(x)$and $\nabla w_i(x)$ — assuming quintic interpolation, W and ∂W will each have 6^d elements per row. As with SKI, we use FFTs to obtain $O(M \log M)$ MVMs with K_{UU}. Because W and ∂W have $O(N6^d)$ and $O(Nd6^d)$nonzero elements, respectively, our MVM complexity is $O(Nd6^d + M \log M)$.

4.5.2 D-SKIP

Several common kernels are *separable*, i.e., they can be expressed as products of one-dimensional kernels. Assuming a compatible approximation scheme, this structure is

inherited by the SKI approximation for the kernel matrix without derivatives,

$$K \approx (W_1 K_1 W_1^T) \odot (W_2 K_2 W_2^T) \odot \ldots \odot (W_d K_d W_d^T), \tag{4.33}$$

where $A \odot B$ denotes the Hadamard product of matrices A and B with the same dimensions, and W_j and K_j denote the SKI interpolation and inducing point grid matrices in the j-th coordinate direction. The same Hadamard product structure applies to the kernel matrix with derivatives; for example, for $d = 2$,

$$K^{\nabla} \approx \begin{bmatrix} W_1 K_1 W_1^T & W_1 K_1 \partial W_1^T & W_1 K_1 W_1^T \\ \partial W_1 K_1 W_1^T & \partial W_1 K_1 \partial W_1^T & \partial W_1 K_1 W_1^T \\ W_1 K_1 W_1^T & W_1 K_1 \partial W_1^T & W_1 K_1 W_1^T \end{bmatrix} \odot \begin{bmatrix} W_2 K_2 W_2^T & W_2 K_2 W_2^T & W_2 K_2 \partial W_2^T \\ W_2 K_2 W_2^T & W_2 K_2 W_2^T & W_2 K_2 \partial W_2^T \\ \partial W_2 K_2 W_2^T & \partial W_2 K_2 W_2^T & \partial W_2 K_2 \partial W_2^T \end{bmatrix}. \tag{4.34}$$

Eq. (4.34) expresses K^{∇} as a Hadamard product of one dimensional kernel matrices. Following this approximation, we apply the SKIP reduction [66] and use Lanczos to further approximate Eq. (4.34) as $(Q_1 \Gamma_1 Q_1^T) \odot (Q_2 \Gamma_2 Q_2^T)$. This can be used for fast MVMs with the kernel matrix. Applied to kernel matrices with derivatives, we call this approach D-SKIP.

Constructing the D-SKIP kernel costs $O(d^2(N + M \log M + r^3 N \log d))$, and each MVM costs $O(dr^2 N)$ flops where r is the effective rank of the kernel at each step (rank of the Lanczos decomposition). We achieve high accuracy with $r \ll N$.

4.5.3 Preconditioning

Recent work [43] has explored several preconditioners for exact kernel matrices. We have had success with preconditioners of the form $H = \sigma^2 I + FF^T$ where $K^{\nabla} \approx FF^T$ with $F \in \mathbb{R}^{N \times r}$. Solving with the Sherman-Morrison-Woodbury formula (*a.k.a* the matrix inversion lemma) is inaccurate for small σ; we use the more stable formula

$H^{-1}b = \sigma^{-2}(f - Q_1(Q_1^T f))$ where Q_1 is computed in $O(r^2 N)$ time by the economy QR factorization

$$\begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} = \begin{bmatrix} F \\ \sigma I \end{bmatrix} R. \tag{4.35}$$

In our experiments with solvers for D-SKI and D-SKIP, we have found that a truncated pivoted Cholesky factorization, $K^{\nabla} \approx (\Pi L)(\Pi L)^T$ works well for the low-rank factorization. Computing the pivoted Cholesky factorization is cheaper than MVM-based preconditioners such as Lanczos or truncated eigendecompositions as it only requires the diagonal and the ability to form the rows where pivots are selected. Pivoted Cholesky is a natural choice when inducing point methods are applied as the pivoting can itself be viewed as an inducing point method where the most important information is selected to construct a low-rank preconditioner [81]. The D-SKI diagonal can be formed in $O(Nd6^d)$ flops while rows cost $O(Nd6^d + M)$ flops; for D-SKIP both the diagonal and the rows can be formed in $O(Nd)$ flops.

4.5.4 Dimensionality Reduction

In many high-dimensional function approximation problems, only a few directions are relevant. That is, if $f : \mathbb{R}^D \to \mathbb{R}$ is a function to be approximated, there is often a matrix P with $d < D$ orthonormal columns spanning an *active subspace* of $\mathbb{R}^D$ such that $f(x) \approx f(PP^T x)$ for all x in some domain Ω of interest [41]. The optimal subspace is given by the dominant eigenvectors of the covariance matrix $C = \int_\Omega \nabla f(x) \nabla f(x)^T \, dx$, generally estimated by Monte Carlo integration. Once the subspace is determined, the function can be approximated through a GP on the reduced space, i.e., we replace the original kernel $k(x, x')$ with a new kernel $\check{k}(x, x') = k(P^T x, P^T x')$. Because we assume gradient information, dimensionality reduction based on active subspaces is a natural

pre-processing phase before applying D-SKI and D-SKIP.

4.6 Experiments: Derivative-Free GPs

We test our stochastic trace estimator with both Chebyshev and Lanczos approximation schemes on a variety of applications. Throughout we use the SKI method [189] of Eq. (4.7) for fast MVMs. We find that the Lanczos and surrogate methods are able to do kernel recovery and inference significantly faster and more accurately than competing methods.

4.6.1 Natural Sound Modeling

Here we consider the natural sound benchmark in [189], shown in Fig. 4.2a. Our goal is to recover contiguous missing regions in a waveform with $N = 59,306$ training points. We exploit Toeplitz structure in the K_{UU} matrix of our SKI approximate kernel for accelerated MVMs.

The experiment in [189] only considered scalable inference and prediction, but not hyper-parameter learning, since the scaled eigenvalue approach requires all the eigenvalues for an $M \times M$ Toeplitz matrix, which can be computationally prohibitive with cost $O(M^2)$. However, evaluating the marginal likelihood on this training set is not an obstacle for Lanczos and Chebyshev since we can use fast MVMs with the SKI approximation at a cost of $O(N + M \log M)$.

In Fig. 4.2b, we show how Lanczos, Chebyshev and surrogate approaches scale with the number of inducing points M compared to the scaled eigenvalue method and FITC.

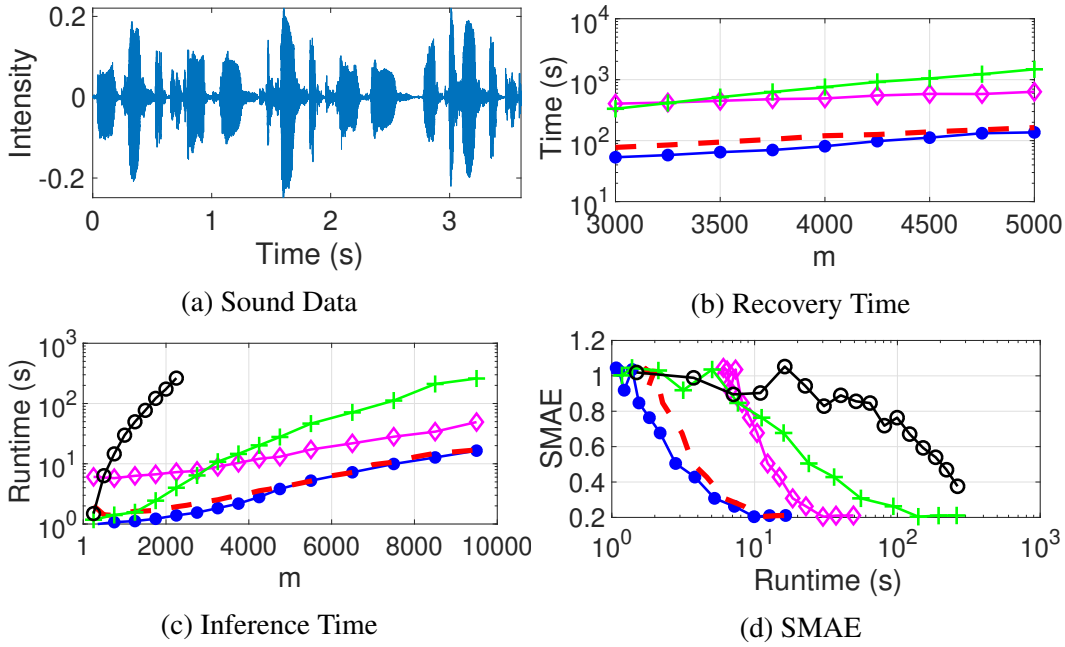
We use 5 probe vectors and 25 iterations for Lanczos, both when building the surrogate and for hyper-parameter learning with Lanczos. We also use 5 probe vectors for Chebyshev and 100 moments. Fig. 4.2b shows the runtime of the hyper-parameter learning phase for different numbers of inducing points M, where Lanczos and the surrogate are clearly more efficient than scaled eigenvalues and Chebyshev. For hyper-parameter learning, FITC took several hours to run, compared to minutes for the alternatives; we therefore exclude FITC from Fig. 4.2b. Fig. 4.2c shows the time to do inference on the 691 test points,while 4.2d shows the standardized mean absolute error (SMAE) on the same test points. As expected, Lanczos and surrogate make accurate predictions much faster than Chebyshev, scaled eigenvalues, and FITC. In short, Lanczos and the surrogate approach are much faster than alternatives for hyper-parameter learning with a large number of inducing points and training points.

4.6.2 Daily Precipitation Prediction

This experiment involves precipitation data from the year of 2010 collected from around 5500 weather stations in the US[1]. The hourly precipitation data is preprocessed into daily data if full information of the day is available. The dataset has $628,474$ entries in terms of precipitation per day given the date, longitude and latitude. We randomly select $100,000$ data points as test points and use the remaining points for training. We then perform hyper-parameter learning and prediction with the RBF kernel, using Lanczos, scaled eigenvalues, and exact methods.

For Lanczos and scaled eigenvalues, we optimize the hyper-parameters on the subset of data for January 2010, with an induced grid of 100 points per spatial dimension and 300 in the temporal dimension. Due to memory constraints we only use a subset

[1] `https://catalog.data.gov/dataset/u-s-hourly-precipitation-data`



(a) Sound Data

(b) Recovery Time

(c) Inference Time

(d) SMAE

Figure 4.2: Sound modeling using 59,306 training points and 691 test points. The intensity of the time series can be seen in (a). Train time for RBF kernel hyper-parameters is in (b) and the time for inference is in (c). The standardized mean absolute error (SMAE) as a function of time for an evaluation of the marginal likelihood and all derivatives is shown in (d). Surrogate is (——), Lanczos is (- - -), Chebyshev is (— ◇ —), scaled eigenvalues is (— + —), and FITC is (— o —).

of 12,000 entries for training with the exact method. While scaled eigenvalues can perform well when fast eigendecompositions are possible, as in this experiment, Lanczos nonetheless still runs faster and with slightly lower mean square error (MSE).

Table 4.1: Prediction Comparison for the Daily Precipitation Data [α].

Method	#Training Pts	#Induced Pts	MSE	Time [min]
Lanczos	528k	3M	0.613	14.3
Scaled-Eig	528k	3M	0.621	15.9
Exact	12k	-	0.903	11.8

[α] The columns are the number of training points, number of induced grid points, mean squared error, and inference time.

Incidentally, we are able to use 3 *million* inducing points in Lanczos and scaled

eigenvalues, which is enabled by the SKI representation [189] of covariance matrices, for a a very accurate approximation. This number of inducing points M is unprecedented for typical alternatives which scale as $O(M^3)$.

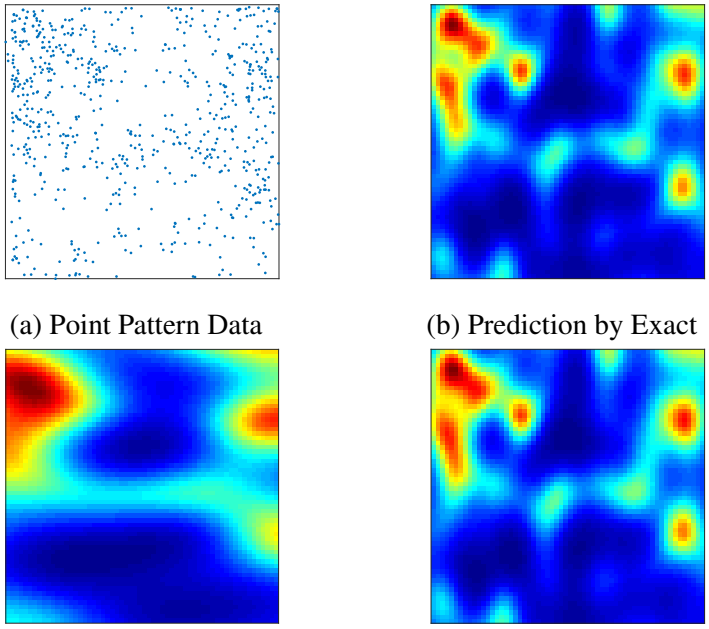
4.6.3 Hickory Data

In this experiment, we apply Lanczos to the log-Gaussian Cox process model with a Laplace approximation for the posterior distribution. We use the RBF kernel and the Poisson likelihood in our model. The scaled eigenvalue method does not apply directly to non-Gaussian likelihoods; we thus applied the scaled eigenvalue method in [189] in conjunction with the Fiedler bound in [64] for the scaled eigenvalue comparison. Indeed, a key advantage of the Lanczos approach is that it can be applied whenever fast MVMs are available, which means no additional approximations such as the Fiedler bound are required for non-Gaussian likelihoods.

This dataset, which comes from the R package `spatstat`, is a point pattern of 703 hickory trees in a forest in Michigan. We discretize the area into a 60×60 grid and fit our model with exact, scaled eigenvalues, and Lanczos. We see in Table 4.2 that Lanczos recovers hyper-parameters that are much closer to the exact values than the scaled eigenvalue approach. Fig. 4.3 shows that the predictions by Lanczos are also indistinguishable from the exact computation.

Table 4.2: Hyperparameters Recovered on the Hickory Dataset.

Method	s_f	ℓ_1	ℓ_2	$-\log p(y\|\theta)$	Time [s]
Exact	0.696	0.063	0.085	1827.56	465.9
Lanczos	0.693	0.066	0.096	1828.07	21.4
Scaled-Eig	0.543	0.237	0.112	1851.69	2.5



(a) Point Pattern Data (b) Prediction by Exact

(c) Scaled-Eig (d) Lanczos

Figure 4.3: Predictions by exact, scaled eigenvalues, and Lanczos on the Hickory dataset.

4.6.4 Crime Prediction

In this experiment, we apply Lanczos with the spectral mixture kernel to the crime forecasting problem considered in [64]. This dataset consists of $233,088$ incidents of assault in Chicago from January 1, 2004 to December 31, 2013. We use the first 8 years for training and attempt to predict the crime rate for the last 2 years. For the spatial dimensions, we use the log-Gaussian Cox process model, with the Matérn-5/2 kernel, the negative binomial likelihood, and the Laplace approximation for the posterior. We use a spectral mixture kernel with 20 components and an extra constant component for the temporal dimension. We discretize the data into a 17×26 spatial grid corresponding to 1 mile $\times$1 mile grid cells. In the temporal dimension we sum our data by weeks for a total of 522 weeks. After removing the cells that are outside Chicago, we have a total of $157,644$ observations.

The results for Lanczos and scaled eigenvalues (in conjunction with the Fiedler bound due to the non-Gaussian likelihood) can be seen in Table 4.3. The Lanczos method used 5 Hutchinson probe vectors and 30 Lanczos steps. For both methods we allow 100 iterations of LBFGS to recover hyper-parameters and we often observe early convergence. While the root mean square error (RMSE) for Lanczos and scaled eigenvalues happen to be close on this example, the recovered hyper-parameters using scaled eigenvalues are very different than for Lanczos. For example, the scaled eigenvalue method learns much larger σ^2 than Lanczos, indicating model misspecification. In general, as the data become increasingly non-Gaussian the Fiedler bound (used for fast scaled eigenvalues on non-Gaussian likelihoods) will become increasingly misspecified, while Lanczos will be unaffected.

Table 4.3: Hyperparameters Recovered, Recovery Time and RMSE for Lanczos and Scaled Eigenvalues on the Chicago Assault Data$^{\alpha}$.

Method	ℓ_1	ℓ_2	σ^2	$T_{recovery}$[s]	$T_{prediction}$[s]	$RMSE_{train}$	$RMSE_{test}$
Lanczos	0.65	0.67	69.72	264	10.30	1.17	1.33
Scaled-Eig	0.32	0.10	191.17	67	3.75	1.19	1.36

$^{\alpha}$ ℓ_1 and ℓ_2 are the length scales in spatial dimensions. σ^2 is the noise level. $T_{recovery}$ is the time for recovering hyper -parameters. $T_{prediction}$ is the time for prediction at all $157,644$ observations, including training and testing.

4.6.5 Deep Kernel Learning

To handle high-dimensional datasets, we bring our methods into the deep kernel learning framework [191] by replacing the final layer of a pre-trained deep neural network (DNN) with a GP. This experiment uses the gas sensor dataset from the UCI machine learning repository. It has 2565 instances with 128 dimensions. We pre-train a DNN, then attach a Gaussian process with RBF kernels to the two-dimensional output of the second-to-last layer. We then further train all parameters of the resulting kernel, *including* the weights

of the DNN, through the GP marginal likelihood. In this example, Lanczos and the scaled eigenvalue approach perform similarly well. Nonetheless, we see that Lanczos can effectively be used with SKI on a high dimensional problem to train hundreds of thousands of kernel parameters.

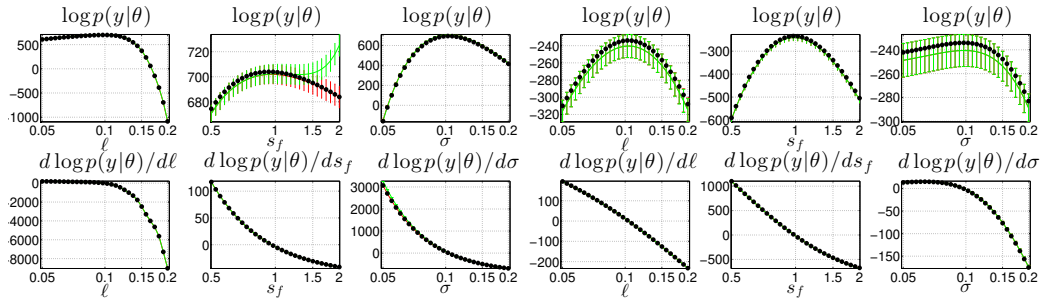
Table 4.4: Prediction RMSE and Per Training Iteration Runtime.

Method	DNN	Lanczos	Scaled-Eig
RMSE	0.1366 ± 0.0387	0.1053 ± 0.0248	0.1045 ± 0.0228
Time [s]	0.4438	2.0680	1.6320

4.6.6 1D Cross-section Plots

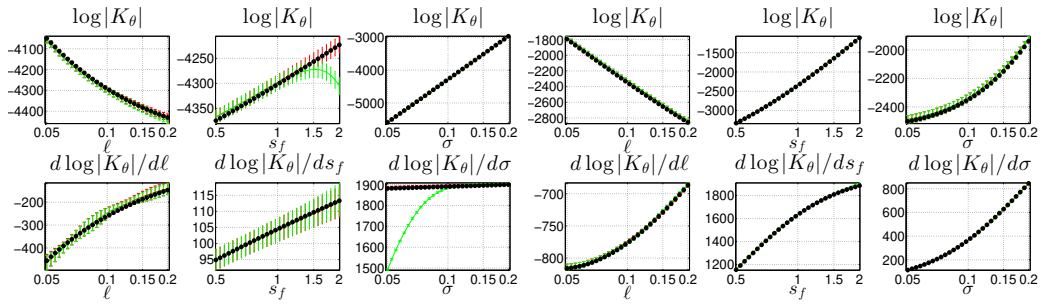
In this experiment we compare the accuracy of Lanczos and Chebyshev for 1-dimensional perturbations of a set of true hyper-parameters, and demonstrate how critical it is to use diagonal replacement for some approximate kernels. We choose the true hyper-parameters to be $(\ell, s_f, \sigma) = (0.1, 1, 0.1)$ and consider two different types of datasets. The first dataset consists of 1000 equally spaced points in the interval $[0, 4]$ in which case the kernel matrix of a stationary kernel is Toeplitz and we can make use of fast matrix-vector multiplication. The second dataset consists of 1000 data points drawn independently from a $U(0, 4)$ distribution. We use SKI with cubic interpolation to construct an approximate kernel based on 1000 equally spaced points. The function values are drawn from a GP with the true hyper-parameters, for both the true and approximate kernel. We use 250 iterations for Lanczos and 250 Chebyshev moments in order to assure convergence of both methods. The results for the first dataset with the RBF and Matérn kernels can be seen in Figs. 4.4a to 4.4d. The results for the second dataset with the SKI kernel can be seen in Figs. 4.5a to 4.5d.

Lanczos yields an excellent approximation to the log determinant and its derivatives



(a) Log marginal likelihood for the RBF kernel

(b) Log marginal likelihood for the Matérn kernel

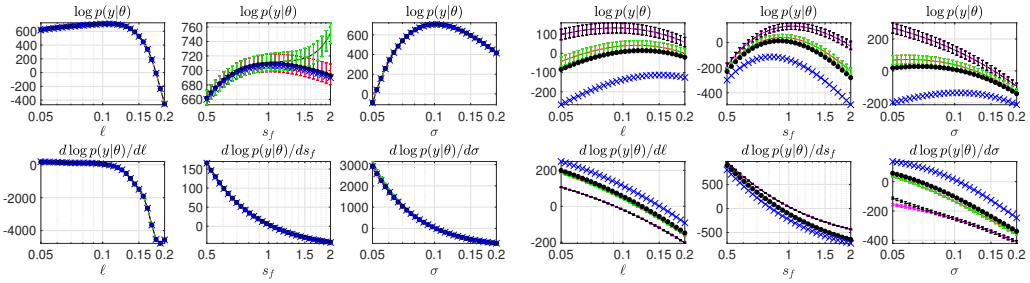


(c) Log determinant for the RBF kernel

(d) Log determinant for the Matérn kernel

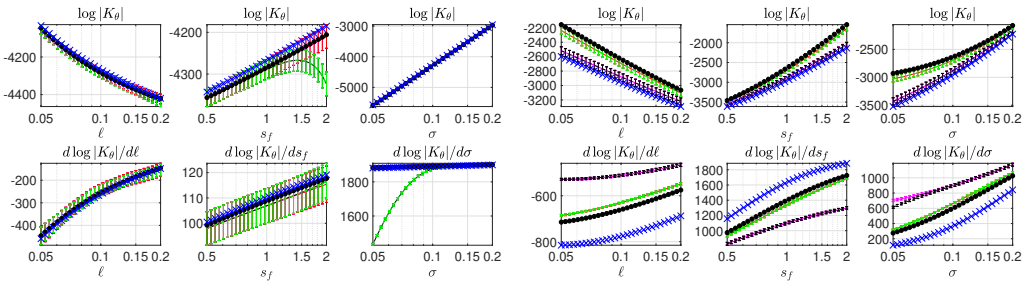
Figure 4.4: 1-dimensional perturbations for the exact RBF and Matérn-1/2 kernel where the data is 1000 equally spaced points in the interval $[0, 4]$. The exact values are (•), Lanczos is (——), Chebyshev is (——). The error bars of Lanczos and Chebyshev are 1 standard deviation and were computed from 10 runs with different probe vectors

for both the exact and the approximate kernels, while Chebyshev struggles with large values of s_f and small values of σ on the exact and approximate RBF kernel. This is expected since Chebyshev has issues with the singularity at zero while Lanczos has large quadrature weights close to zero to compensate for this singularity. The scaled eigenvalue method has issues with the approximate Matérn-1/2 kernel.



(a) Log marginal likelihood for the RBF kernel

(b) Log marginal likelihood for the Matérn kernel



(c) Log determinant for the RBF kernel

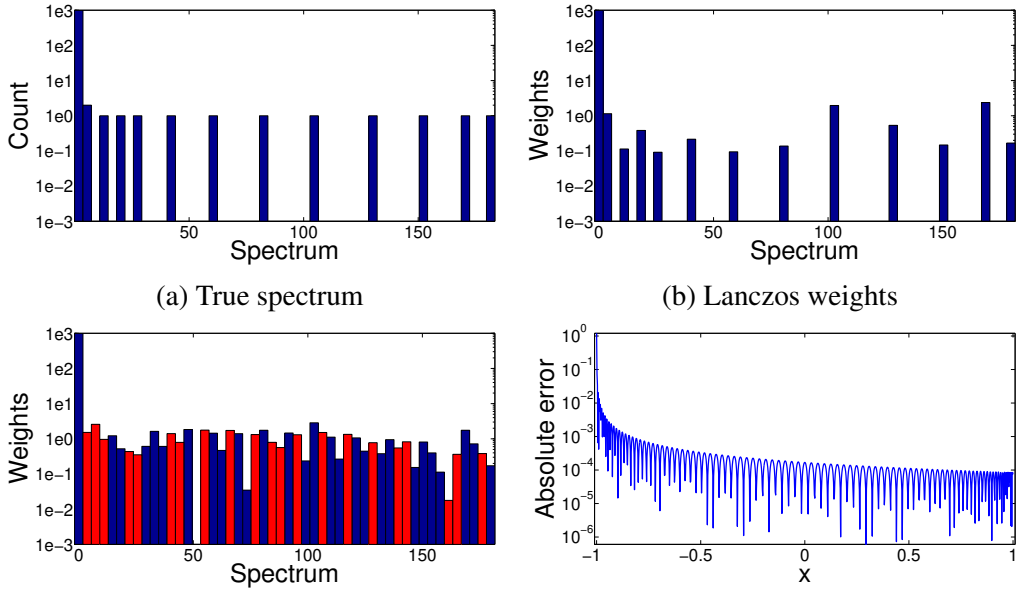
(d) Log determinant for the Matérn kernel

Figure 4.5: 1-dimensional perturbations with the SKI (cubic) approximations of the RBF and Matérn-1/2 kernel where the data is 1000 points drawn from $\mathcal{N}(0, 2)$. The exact values are (•), Lanczos with diagonal replacement is (——), Chebyshev with diagonal replacement is (——), Lanczos without diagonal replacement is (——), Chebyshev without diagonal replacement is (——), and scaled eigenvalues is (×). Diagonal replacement makes no perceptual difference for the RBF kernel so the lines are overlapping in this case. The error bars of Lanczos and Chebyshev are 1 standard deviation and were computed from 10 runs with different probe vectors

4.6.7 Why Lanczos is Better Than Chebyshev

In this experiment, we study the performance advantage of Lanczos over Chebyshev. Fig. 4.6 shows that the Ritz values of Lanczos quickly converge to the spectrum of the RBF kernel thanks to the absence of interior eigenvalues. The Chebyshev approximation shows the expected equioscillation behavior. More importantly, the Chebyshev approximation for logarithms has its greatest error near zero where the majority of the eigenvalues are, and those also have the heaviest weight in the log determinant.

Another advantage of Lanczos is that it requires minimal knowledge of the spectrum, while Chebyshev needs the extremal eigenvalues for rescaling. In addition, with Lanczos we can get the derivatives with only one MVM per hyper-parameter, while Chebyshev requires an MVM at each iteration, leading to extra computation and memory usage.



(a) True spectrum

(b) Lanczos weights

(c) Chebyshev weights

(d) Chebyshev absolute error

Figure 4.6: A comparison between the true spectrum, the Lanczos weights ($m = 50$), and the Chebyshev weights ($m = 100$) for the RBF kernel with $\ell = 0.3$, $s_f = 1$, and $\sigma = 0.1$. All weights and counts are on a log-scale so that they are easier to compare. Blue bars correspond to positive weights while red bars correspond to negative weights.

4.6.8 Importance of Diagonal Correction

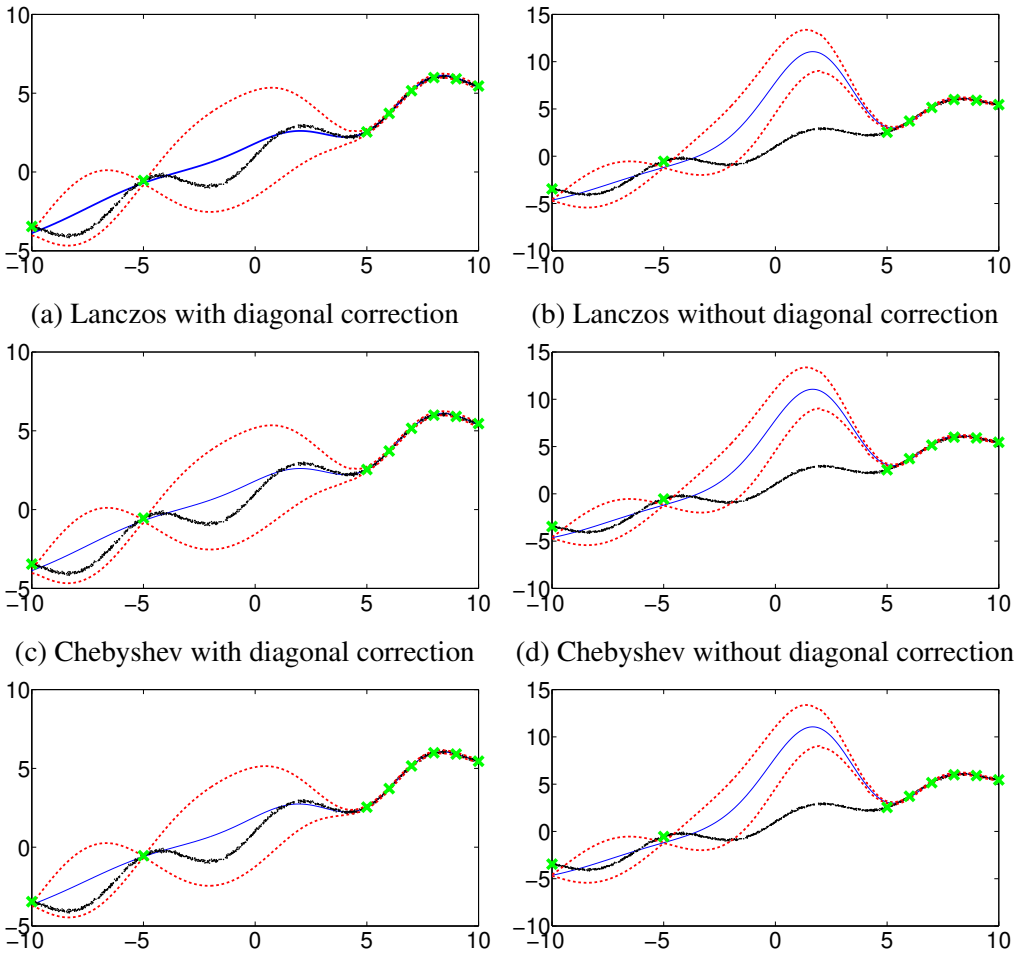
This experiment shows that diagonal correction of the approximate kernel can be very important. Diagonal correction cannot be used efficiently for some methods, such as the scaled eigenvalue method, and this may hurt its predictive performance. Our experiment is similar to [154]. We generate 1000 uniformly distributed points in the interval

$[-10, 10]$, and we choose a small number of inducing points in such a way that there is a large chunk of the interval where there is no inducing point. We are interested in the behavior of the predictive uncertainties on this subinterval. The function values are given by $f(x) = 1 + x/2 + \sin(x)$ and normally distributed noise with standard deviation 0.05 is added to the function values. We find the optimal hyper-parameters of the Matérn-3/2 using the exact method and use these hyper-parameters to make predictions with Lanczos, Chebyshev, FITC, and the scaled eigenvalue method. We consider Lanczos both with and without diagonal correction in order to see how this affects the predictions. The results can be seen in Figure 4.7.

It is clear that Lanczos and Chebyshev are too confident in the predictive mean when diagonal correction is not used, while the predictive uncertainties agree well with FITC when diagonal correction is used. The scaled eigenvalue method cannot be used efficiently with diagonal correction and we see that this leads to predictions similar to Lanczos and Chebyshev without diagonal correction. The flexibility of being able to use diagonal correction with Lanczos and Chebyshev makes these approaches very appealing.

4.6.9 Surrogate Log Determinant Approximation

The point of this experiment is to illustrate how accurate the level-curves of the surrogate model are compared to the level-curves of the true log determinant. We consider the RBF and the Matérn-3/2 kernels and the same datasets that we considered in 4.6.6. We fix $s_f = 1$ and study how the level curves compare when we vary ℓ and σ. Building the surrogate with all three hyper-parameters produces similar results, but requires more design points. We use 50 design points to construct a cubic RBF with a linear tail.



(a) Lanczos with diagonal correction

(b) Lanczos without diagonal correction

(c) Chebyshev with diagonal correction

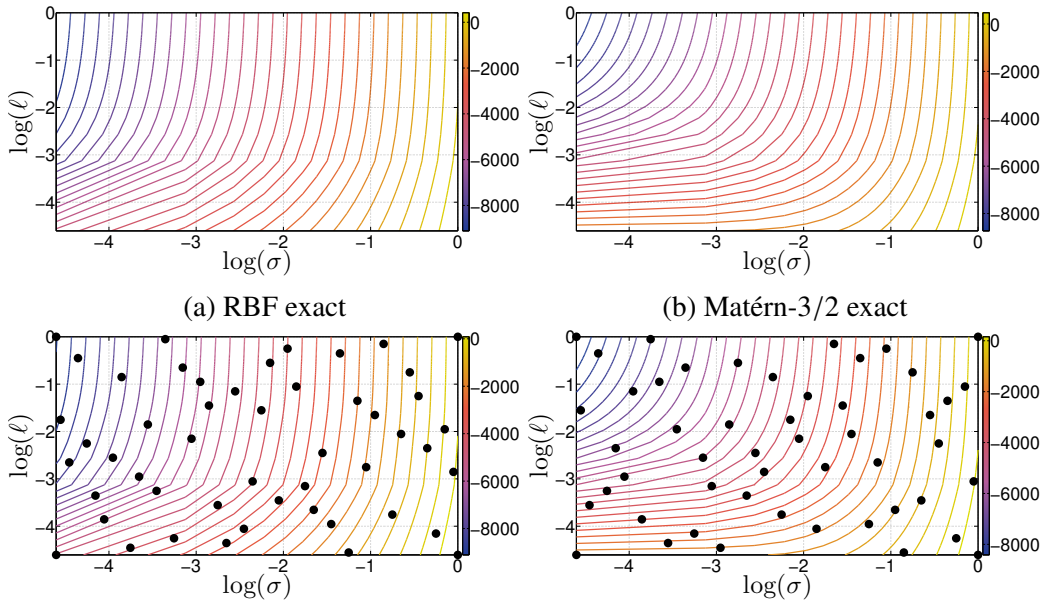
(d) Chebyshev without diagonal correction

(e) FITC

(f) Scaled eigenvalue method

Figure 4.7: Example that shows how important diagonal correction can be for some kernels. The Matérn-3/2 kernel was used to fit the data given by the black dots. This data was generated from the function $f(x) = 1 + x/2 + \sin(x)$ to which we added normally distributed noise with standard deviation 0.05. We used the exact method to find the optimal hyper-parameters and used these hyper-parameters to study the different behavior of the predictive uncertainties when the inducing points are given by the green crosses. The solid blue line is the predictive mean and the dotted red lines shows a confidence interval of two standard deviations.

The values of the log determinant and its derivatives are computed with Lanczos. It is clear from Fig. 4.8 that the surrogate model does a good job approximating the log determinant for both kernels.



(a) RBF exact

(b) Matérn-3/2 exact

(c) RBF surrogate

(d) Matérn-3/2 surrogate

Figure 4.8: Level curves of the exact and surrogate approximation of the log determinant as a function of ℓ and σ for the RBF and Matérn-3/2 kernels. We used $s_f = 1$ and the dataset consisted of 1000 equally spaced points in the interval $[0, 4]$. The surrogate model was constructed from the points shown with (•) and the log determinant values were computed using stochastic Lanczos.

4.6.10 Kernel Hyper-parameter Recovery

This experiments tests how well we can recover hyper-parameters from data generated from a GP. We compare Chebyshev, Lanczos, the surrogate, the scaled eigenvalue method, and FITC. We consider a dataset of 5000 points generated from a $\mathcal{N}(0, 2)$ distribution. We use SKI with cubic interpolation and a total of 2000 inducing points for Lanczos, Chebyshev, and then scaled eigenvalue method. FITC was used with 750

equally spaced points because it has a longer runtime as a function of the number of inducing points. We consider the RBF kernel and the Matérn-3/2 kernel and sample from a GP with ground truth parameters $(\ell, s_f, \sigma) = (0.01, 0.5, 0.05)$. The GPs for which we try to recover the hyper-parameters were generated from the original kernel. It is important to emphasize that there are two sources of errors present: the error from the kernel approximation errors and the stochastic error from Lanczos and Chebyshev. We saw in Figs. 4.4 and 4.5 that the stochastic error for Lanczos is relatively small, so this follow-up experiment helps us understand how Lanczos is influenced by the error incurred from an approximate kernel. We show the true log marginal likelihood, the recovered hyper-parameters, and the run-time in Table 4.5.

It is clear from Table 4.5 that most methods are able to recover parameters close to the ground truth for the RBF kernel. The results are more interesting for the Matérn-3/2 kernel where FITC struggles and the parameters recovered by FITC have a value of the log marginal likelihood that is much worse than the other methods.

4.7 Experiments: GPs with Derivative Information

The experiments in this section use the squared exponential (SE) kernel, which has product structure and can be used with D-SKIP; and the spline kernel, to which D-SKIP does not directly apply. We use these kernels in tandem with D-SKI and D-SKIP to achieve the fast MVMs derived in Section 4.5. We write D-SE to denote the exact SE kernel with derivatives.

Table 4.5: Hyper-parameter recovery for the RBF and Matérn-3/2 kernels[α].

		RBF	Matérn 3/2
True	$-\log p(y\|\theta)$	−6.22e3	−4.91e3
	Hypers	(0.01, 0.5, 0.05)	(0.01, 0.5, 0.05)
Exact	$-\log p(y\|\theta)$	−6.23e3	−4.91e3
	Hypers	(1.01e−2, 4.81e−1, 5.03e−2)	(9.63e−3, 4.87e−1, 4.96e−2)
	Time (s)	368.9	466.7
Lanczos	$-\log p(y\|\theta)$	−6.22e3	−4.86e3
	Hypers	(1.00e−2, 4.77e−1, 5.03e−2)	(1.04e−2, 4.87e−1, 4.67e−2)
	Time (s)	66.2	133.4
Chebyshev	$-\log p(y\|\theta)$	−6.23e3	−4.81e3
	Hypers	(9.84e−3, 4.85e−1, 5.12e−2)	(1.11e−2, 4.66e−1, 5.78e−2)
	Time (s)	110.3	173.3
Surrogate	$-\log p(y\|\theta)$	−6.22e3	−4.86e3
	Hypers	(1.01e−2, 4.88e−1, 4.85e−2)	(1.02e−2, 4.80e−1, 4.66e−2)
	Time (s)	48.2	44.3
Scaled Eig	$-\log p(y\|\theta)$	−6.22e3	−4.71e3
	Hypers	(1.04e−2, 4.52e−1, 5.14e−2)	(1.13e−2, 4.53e−1, 6.37e−2)
	Time (s)	90.2	127.3
FITC	$-\log p(y\|\theta)$	−6.22e3	−4.11e3
	Hypers	(1.03e−2, 4.90e−1, 5.07e−2)	(1.34e−2, 5.22e−1, 8.91e−2)
	Time (s)	86.6	136.9

[α] The data was generated from 5000 normally distributed points. Lanczos, surrogate, and scaled eigenvalues all used 2000 inducing points while FITC used 750. These numbers where chosen to make their run times close to equal. Diagonal correction was applied to the Matérn-3/2 approximate kernel. The value of the log marginal likelihood was was computed from the exact kernel and shows the value of the hyper-parameters recovered by each method. We ran Lanczos 5 times and averaged the values.

4.7.1 Approximation Benchmark

D-SKI and D-SKIP with the SE kernel approximate the original kernel well, both in terms of MVM accuracy and spectral profile. Comparing D-SKI and D-SKIP to their exact counterparts in Figure 4.9, we see their matrix entries are very close (leading to MVM accuracy near 10^{-5}), and their spectral profiles are indistinguishable. The same

is true with the spline kernel.

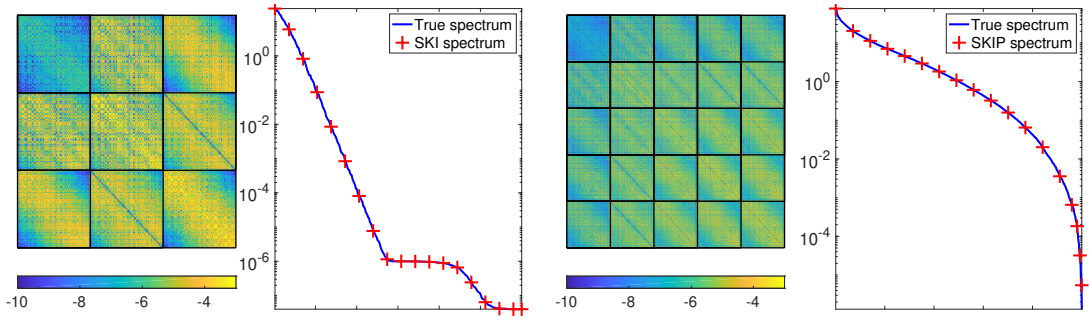


Figure 4.9: (Left two images) $\log_{10}$ error in SKI approximation and comparison to the exact spectrum.(Right two images) $\log_{10}$ error in SKIP approximation and comparison to the exact spectrum.

Additionally, scaling tests in Fig. 4.10 verify the predicted complexity of D-SKI and D-SKIP. We show the relative fitting accuracy of SE, SKI, D-SE, and D-SKI on some standard test functions in Table 4.6.

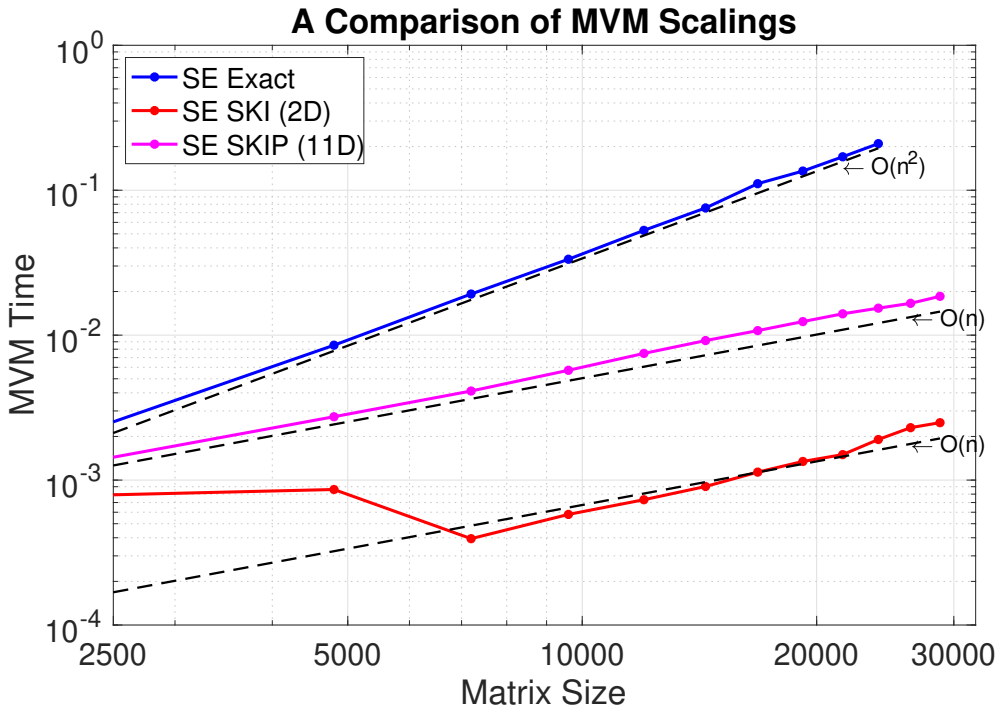


Figure 4.10: Scaling tests for D-SKI in two dimensions and D-SKIP in 11 dimensions. D-SKIP uses fewer data points for identical matrix sizes.

Table 4.6: Relative RMSE on Test Functions Using SKI and Derivatives [α].

	Branin	Franke	Sine Norm	Sixhump	StyTang	Hart3
SE	6.02e-3	8.73e-3	8.64e-3	6.44e-3	4.49e-3	1.30e-2
SKI	3.97e-3	5.51e-3	5.37e-3	5.11e-3	2.25e-3	8.59e-3
D-SE	1.83e-3	1.59e-3	3.33e-3	1.05e-3	1.00e-3	3.17e-3
D-SKI	1.03e-3	4.06e-4	1.32e-3	5.66e-4	5.22e-4	1.67e-3

[α] Relative RMSE error measured on 10000 testing points. Test functions from [170] includes five 2D functions (Branin, Franke, Sine Norm, Six-hump, and Styblinski-Tang) and one 3D function (Hartman). We train the SE kernel on 4000 points, the D-SE kernel on $4000/(d+1)$ points, and SKI and D-SKI with SE kernel on 10000 points to achieve comparable runtimes between methods.

4.7.2 Preconditioning

We discover that preconditioning is crucial for the convergence of iterative solvers using approximation schemes such as D-SKI and D-SKIP. To illustrate the performance of conjugate gradient method with and without the above-mentioned truncated pivoted Cholesky preconditioner, we test the D-SKI on 2D Franke function with 2000 data points, and D-SKIP on 5D Friedman function with 1000 data points. In both cases, we compute a pivoted Cholesky decomposition truncated at rank 100 for preconditioning, and the number of steps it takes for CG/PCG to converge is demonstrated in Fig. 4.11 below. It is clear that preconditioning universally and significantly reduces the number of steps required for convergence.

4.7.3 Dimensionality Reduction

We apply active subspace pre-processing to the 20 dimensional Welsh test function in [24]. The top six eigenvalues of its gradient covariance matrix are well separated from the rest as seen in Fig. 4.12a. However, the function is far from smooth when projected

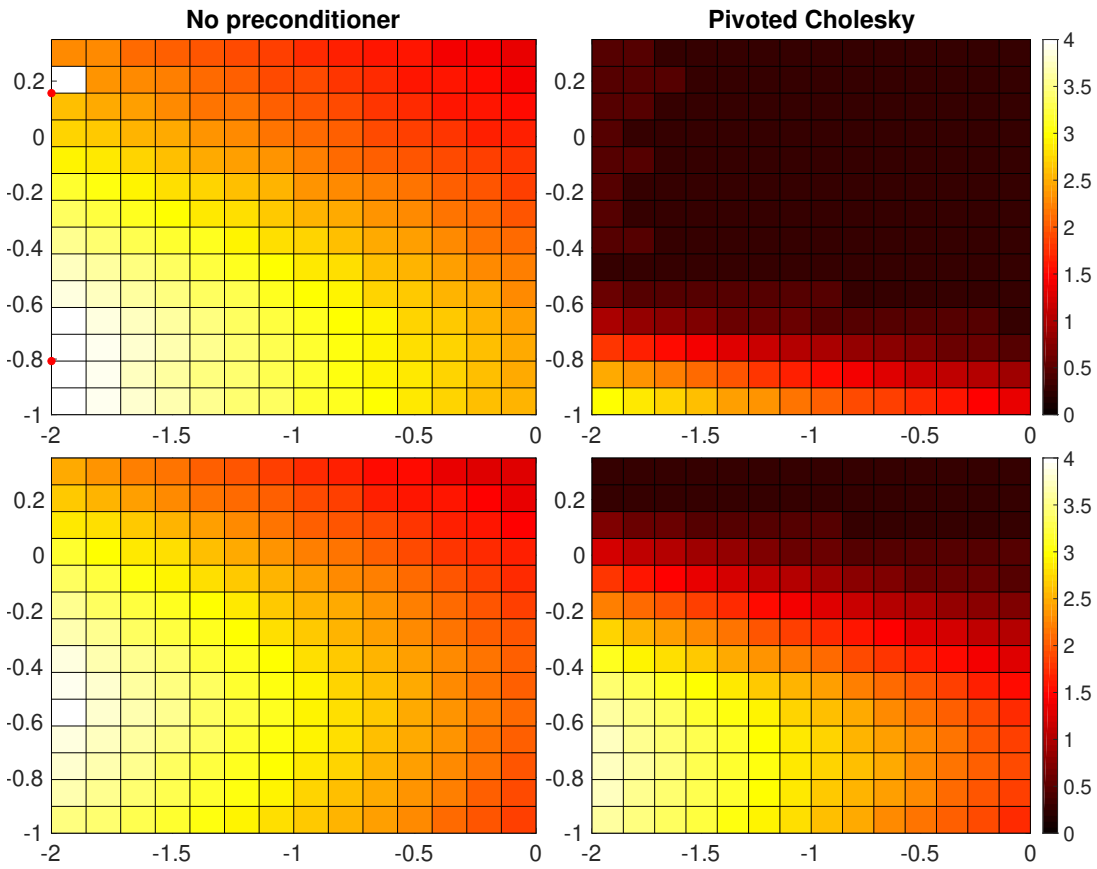
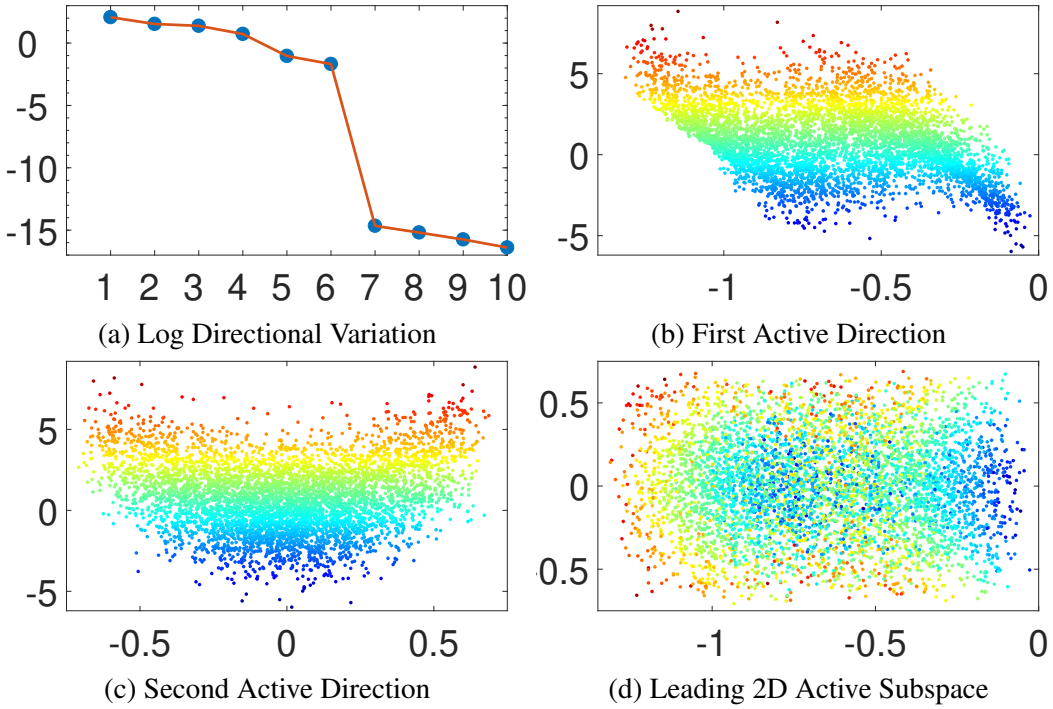


Figure 4.11: The color shows $\log_{10}$ of the number of iterations to reach a tolerance of 1e−4. The first row compares D-SKI with and without a preconditioner. The second row compares D-SKIP with and without a preconditioner. The red dots represent no convergence. The y-axis shows $\log_{10}(\ell)$ and the x-axis $\log_{10}(\sigma)$ and we used a fixed value of $s = 1$.

onto the leading 1D or 2D active subspace, as Figs. 4.12b to 4.12d indicates, where the color shows the function value.

We therefore apply D-SKI and D-SKIP on the 3D and 6D active subspace, respectively, using 5000 training points, and compare the prediction error against D-SE with 190 training points because of our scaling advantage. Table 4.7 reveals that while the 3D active subspace fails to capture all the variation of the function, the 6D active subspace is able to do so. These properties are demonstrated by the poor prediction of D-SKI in 3D and the excellent prediction of D-SKIP in 6D.



(a) Log Directional Variation

(b) First Active Direction

(c) Second Active Direction

(d) Leading 2D Active Subspace

Figure 4.12: Fig. 4.12a shows the top 10 eigenvalues of the gradient covariance. Welsh is projected onto the first and second active direction in 4.12b and 4.12c. After joining them together, we see in 4.12d that points of different color are highly mixed, indicating a very spiky surface.

Table 4.7: Relative RMSE and SMAE prediction error for Welsh [α].

	D-SE	D-SKI (3D)	D-SKIP (6D)
RMSE	4.900e-02	2.267e-01	3.366e-03
SMAE	4.624e-02	2.073e-01	2.590e-03

[α] The D-SE kernel is trained on $4000/(d+1)$ points, with D-SKI and D-SKIP trained on 5000 points. The 6D active subspace is sufficient to capture the variation of the test function.

4.7.4 Rough Terrain Reconstruction

Rough terrain reconstruction is a key application in robotics [70, 105], autonomous navigation [78], and geostatistics. Through a set of terrain measurements, the problem is to predict the underlying topography of some region. In the first example, we consider

roughly 23 million non-uniformly sampled elevation measurements of Mount St. Helens obtained via LiDAR [40]. We bin the measurements into a 970×950 grid, and downsample to a 120×117 grid. Derivatives are approximated using a finite difference scheme.

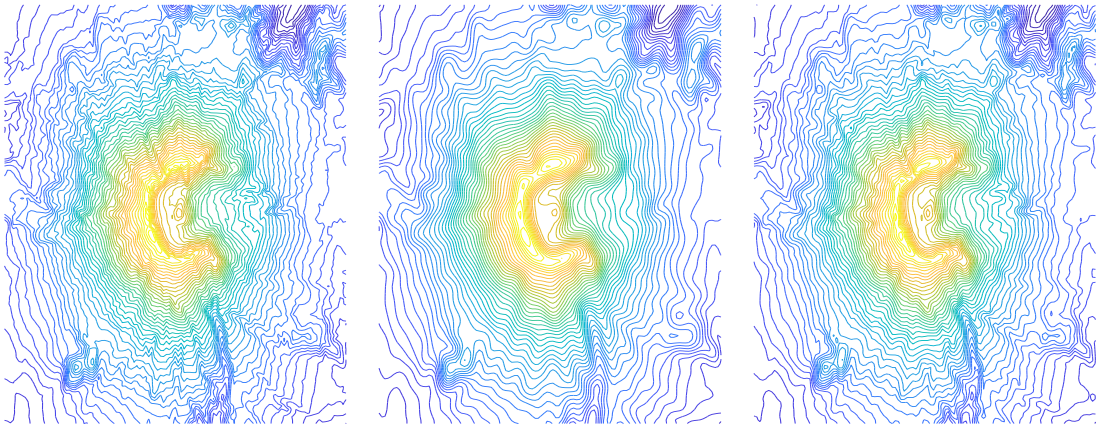


Figure 4.13: On the left is the true elevation map of Mount St. Helens. In the middle is the elevation map calculated with the SKI. On the right is the elevation map calculated with D-SKI.

We randomly select 90% of the grid for training and the remainder for testing. We do not include results for D-SE, as its kernel matrix has dimension roughly 4×10^4. We plot contour maps predicted by SKI and D-SKI in Fig. 4.13 — the latter looks far closer to the ground truth than the former. This is quantified in the following table:

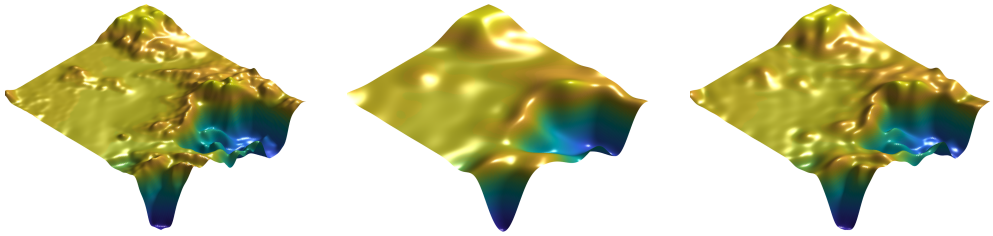
Table 4.8: Hyperparameters Recovered, Recovery SMAE, and Recovery Time for SKI and D-SKI on Mountain St. Helens Data$^{\alpha}$.

	ℓ	s	σ_1	σ_2	$\text{SMAE}_{\text{test}}$	SMAE_{all}	Time[s]
SKI	35.196	207.689	12.865	n.a.	0.0308	0.0357	37.67
D-SKI	12.630	317.825	6.446	2.799	0.0165	0.0254	131.70

$^{\alpha}$ σ_1 and σ_2 are the noise parameters for value and gradient, respectively.

In the second example, the Korean Peninsula elevation and bathymetry dataset [131]

is sampled at a resolution of 12 cells per degree. It has 180×240 entries on a rectangular grid, among which we take 171×231 interior points as the test data a smaller subgrid of 17×23 points as the training data. To reduce data noise, we apply a Gaussian filter with $\sigma_{\text{filter}} = 2$ as a pre-processing step. We observe that the recovered surfaces with SKI and D-SKI highly resemble the respective counterparts with exact computation, and incorporating gradient information enables us to recover more details of the terrain.



(a) Ground Truth (b) SKI (c) D-SKI

Figure 4.14: D-SKI is clearly able to capture more detail in the map than SKI. Note that inclusion of derivative information in this case leads to a negligible increase in calculation time.

Table 4.9: Hyper-parameters Recovered, Recovery SMAE, and Recovery Time for SKI and D-SKI on Korea Peninsula Data.

	ℓ	s	σ	SMAE	Time[s]
SKI	16.786	855.406	184.253	0.1521	10.094
D-SKI	9.181	719.376	29.486	0.0746	11.643

4.7.5 Implicit Surface Reconstruction

Reconstructing surfaces from point cloud data and surface normals is a standard problem in computer vision and graphics. One popular approach is to fit an implicit function that is zero on the surface with gradients equal to the surface normal. Local Hermite RBF interpolation has been considered in prior work [123], but this approach is sensitive to noise. In our experiments, using a GP instead of splining reproduces implicit surfaces

with very high accuracy. In this case, a GP with derivative information is required, as the function values are all zero.

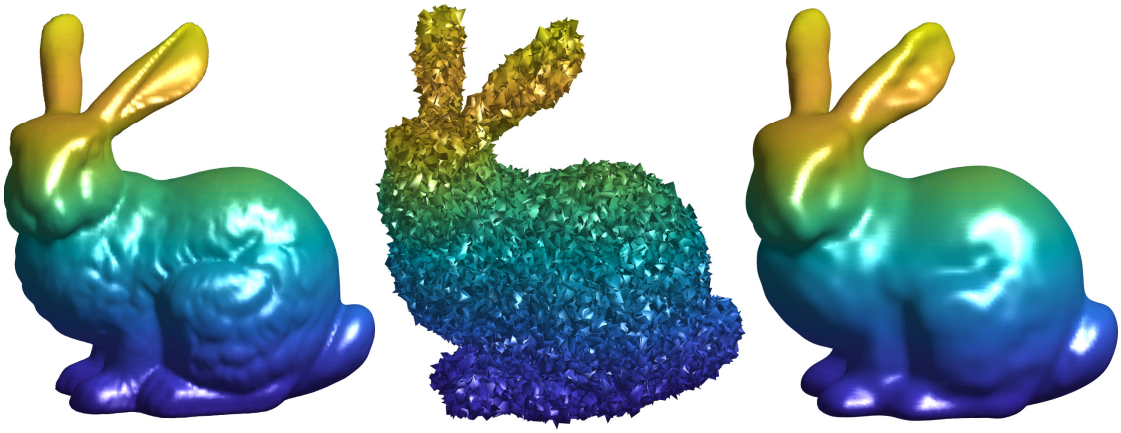


Figure 4.15: (Left) Original surface (Middle) Noisy surface (Right) SKI reconstruction from noisy surface ($s = 0.4$, $\sigma = 0.12$).

In Fig. 4.15, we fit the Stanford bunny using 25000 points and associated normals, leading to a K^{∇} matrix of dimension 10^5, clearly far too large for exact training. We therefore use SKI with the thin-plate spline kernel, with a total of 30 grid points in each dimension. The left image is a ground truth mesh of the underlying point cloud and normals. The middle image shows the same mesh, but with heavily noised points and normals. Using this noisy data, we fit a GP and reconstruct a surface shown in the right image, which looks very close to the original.

4.7.6 Bayesian Optimization with Derivatives

Prior work examines Bayesian optimization (BO) with derivative information in low-dimensional spaces to optimize model hyper-parameters [192]. Wang et al. consider high-dimensional BO (without gradients) with random projections uncovering low-dimensional structure [178]. We propose BO with derivatives and dimensionality reduc-

tion via active subspaces, detailed in Algorithm 4.1.

Algorithm 4.1: BO with derivatives and active subspace learning

while *Budget not exhausted* **do**
  Calculate active subspace projection $P \in \mathbb{R}^{D \times d}$ using sampled gradients
  Optimize acquisition function, $u_{n+1} = \arg\max \mathcal{A}(u)$ with $x_{n+1} = Pu_{n+1}$
  Sample point x_{n+1}, value f_{n+1}, and gradient ∇f_{n+1}
  Update data $\mathcal{D}_{i+1} = \mathcal{D}_i \cup \{x_{n+1}, f_{n+1}, \nabla f_{n+1}\}$
  Update hyper-parameters of GP with gradient defined by kernel $k(P^T x, P^T x')$
end

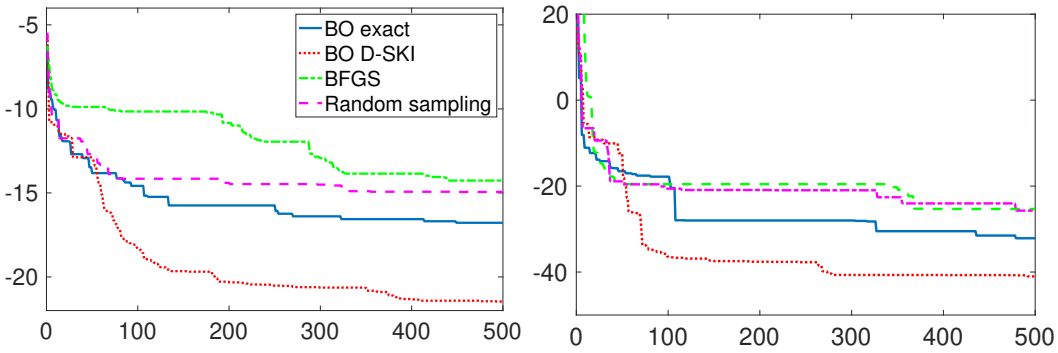
Algorithm 4.1 estimates the active subspace and fits a GP with derivatives in the reduced space. Kernel learning, fitting, and optimization of the acquisition function all occur in this low-dimensional subspace. In our tests, we use the expected improvement (EI) acquisition function, which involves both the mean and predictive variance. We consider two approaches to rapidly evaluate the predictive variance $v(x) = k(x,x) - K_{xX}\tilde{K}^{-1}K_{Xx}$ at a test point x. In the first approach, which provides a biased estimate of the predictive variance, we replace K^{-1} with the preconditioner solve computed by pivoted Cholesky; using the stable QR-based evaluation algorithm, we have

$$v(x) \approx \hat{v}(x) \equiv k(x,x) - \sigma^{-2}(\left\|K_{Xx}\right\|^2 - \left\|Q_1^T K_{Xx}\right\|^2)\,. \tag{4.36}$$

We note that the approximation $\hat{v}(x)$ is always a (small) overestimate of the true predictive variance $v(x)$. In the second approach, we use a randomized estimator as in [22] to compute the predictive variance at many points X' simultaneously, and use the pivoted Cholesky approximation as a control variate to reduce the estimator variance:

$$v_{X'} = \operatorname{diag}(K_{X'X'}) - \mathbb{E}_z\left[z \odot (K_{X'X}\tilde{K}^{-1}K_{XX'}z - K_{X'X}M^{-1}K_{XX'}z)\right] - \hat{v}_{X'}\,. \tag{4.37}$$

The latter approach is unbiased, but gives very noisy estimates unless many probe vectors z are used. Both the pivoted Cholesky approximation to the predictive variance and the randomized estimator resulted in similar optimizer performance in our experiments.



(a) BO on Ackley

(b) BO on Rastrigin

Figure 4.16: In the following experiments, 5D Ackley and 5D Rastrigin are embedded into 50 a dimensional space. We run Algorithm 4.1, comparing it with BO exact, multi-start BFGS, and random sampling. D-SKI with active subspace learning clearly outperforms the other methods.

To test Algorithm 4.1, we mimic the experimental set up in [178]: we minimize the 5D Ackley and 5D Rastrigin test fuctions [170], randomly embedded respectively in $[-10, 15]^{50}$ and $[-4, 5]^{50}$. We fix $d = 2$, and at each iteration pick two directions in the estimated active subspace at random to be our active subspace projection P. We use D-SKI as the kernel and EI as the acquisition function. The results of these experiments are shown in Figs. 4.16a and 4.16b, in which we compare Algorithm 1 to three other baseline methods: BO with EI and no gradients in the original space; multi-start BFGS with full gradients; and random search. In both experiments, the BO variants perform better than the alternatives, and our method outperforms standard BO.

4.8 Conclusion

There are many cases in which fast MVMs can be achieved for kernel matrices, but it is difficult or impossible to efficiently compute a log determinant. We have developed a framework for scalable and accurate estimates of a log determinant and its derivatives

relying only on MVMs. We particularly consider scalable kernel learning, showing the promise of stochastic Lanczos estimation combined with a pre-computed surrogate model. We have shown the scalability and flexibility of our approach through experiments with kernel learning for several real-world data sets using both Gaussian and non-Gaussian likelihoods, and highly parametrized deep kernels.

When derivative data are available, they are a valuable source of information for Gaussian process regression; but inclusion of d extra pieces of information per point naturally leads to new scaling issues. We introduce two methods to deal with these scaling issues: D-SKI and D-SKIP. Both are structured interpolation methods, and the latter also uses kernel product structure. We have also discussed practical details — preconditioning is necessary to guarantee convergence of iterative methods and active subspace calculation reveals low-dimensional structure when gradients are available. We present several experiments with kernel learning, dimensionality reduction, terrain reconstruction, implicit surface fitting, and scalable Bayesian optimization with gradients. For simplicity, these examples all possessed full gradient information; however, our methods trivially extend if only partial gradient information is available.

CHAPTER 5

ROBUST LARGE-VOCABULARY TOPIC MODELING

LIST OF SYMBOLS

Across many data domains, co-occurrence statistics about the joint appearance of objects are powerfully informative. By transforming unsupervised learning problems into decompositions of co-occurrence statistics, spectral inference provides transparent algorithms and optimality guarantees for non-linear dimensionality reduction or latent topic analysis. As object vocabularies grow, however, it becomes rapidly more expensive to store and run inference algorithms on co-occurrence statistics. Current rectification techniques, which preprocess real data in order to overcome model-data mismatch, are even more expensive, as they require iterative projections that destroy sparsity of the co-occurrence data. In this chapter, we propose novel approaches that can simultaneously compress and rectify the co-occurrence statistics, scaling gracefully with the size of vocabulary and the dimension of latent space. We also present new algorithms that are capable of learning latent variables from the compressed statistics without losing visible precision, and verify that they perform comparably to previous approaches on both textual and non-textual data.

5.1 Introduction

Understanding the low-dimensional geometry of noisy and complex data is a fundamental problem of unsupervised learning. Probabilistic models explain data generation processes in terms of low-dimensional latent variables. Inferring a posterior distribution for these latent variables provides us with a compact representation for various exploratory analyses and downstream tasks. However, exact inference is often intractable due to entangled interactions between the latent variables [25, 1, 60, 153]. Variational inference transforms the posterior approximation into an optimization problem over simpler distributions with independent parameters [99, 177, 26], while Markov Chain Monte Carlo enables users to sample from the desired posterior distribution [141, 142, 161]. How-

ever, these likelihood-based methods require numerous iterations without any guarantee beyond local improvement at each step [107].

When the data consists of collections of discrete objects, co-occurrence statistics summarize interactions between objects. Collaborative filtering learns low-dimensional representations of individual items, which are useful for recommendation systems, by explicitly decomposing the co-occurrence of items that are jointly consumed by certain users [111, 116]. Word-vector models learn low-dimensional embeddings of individual words, which encode useful linguistic biases for neural networks, by implicitly decomposing the co-occurrence of words that appear together [151, 115]. If co-occurrence provides a rich enough set of unbiased moments about an underlying generative model, spectral methods can provably learn posterior configurations from co-occurrence information alone, without iterating through individual training examples [7, 4, 89, 2].

However, two major limitations hinder users from taking advantage of spectral inference based on co-occurrence. First, the second-order co-occurrence matrix grows quadratically in the size of the objects (e.g. items, words, products). Pruning these objects is an option, but for a retailer selling millions of products, a low-dimensional representation of a small subset of the products is inadequate. Second, inference quality is poor in real data that does not necessarily follow our computational model. Whereas likelihood-based methods have an intrinsic capability to fit the data to the model despite their mismatch, sample noise can destroy the performance of spectral methods even if the data is synthesized from the model [107]. *Rectification*, a process of projecting empirical co-occurrence onto a set consistent with the geometry of the model, can improve the performance of spectral inference in the face of model mismatch [111]. But running multiple projections dominates overall inference cost even when the vocabulary is small. In addition, the rectification process makes the co-occurrence dense, exacerbating

storage costs when dealing with large vocabularies.

In this chapter, we propose the Epsilon Non-Negative rectification (ENN) and the Low-rank Anchor Word algorithm (LAW). Given a vocabulary of N objects and the user-specified latent dimension K, ENN simultaneously compresses and rectifies the co-occurrence matrix $C \in \mathbb{R}^{N \times N}$ into YY^T with $Y \in \mathbb{R}^{N \times K}$. Each entry of the decompression $(YY^T)_{ij}$ tightly approximates the rectified co-occurrence C^*_{ij} but is allowed to be a tiny negative value above $-\epsilon$. Then LAW learns the latent clusters (e.g., topics of documents or genres of items) and their correlations provided only with Y, guaranteeing the same performance as running the original Anchor Word algorithm on C^* if $YY^T \geq 0$. In contrast, we also formulate the Proximal Alternating Linearized Minimization rectification algorithm (PALM) that approximates the rectified co-occurrence C^* by YY^T with $Y \geq 0$. While the non-negativity of Y in the PALM approach allows LAW to perform exact inference, the fact that PALM enforces more constraints than ENN means that PALM also provides a less faithful approximation to the original co-occurrence data. Our experiments on various textual and non-textual datasets show that ENN learns a high-quality factor Y for which LAW provides results of quality comparable to those based on the full co-occurrence C^*; in contrast, while PALM works comparably in some settings, in others there is a visible loss of accuracy.

We also adopt a randomized algorithm that constructs a low-rank approximation of the full co-occurrence C directly from the raw data. While PALM requires the full co-occurrence, ENN can work directly with the low-rank initialization, eliminating the need to ever store a full co-occurrence matrix. Note also that the second-order methods rely on the *separability assumption*,[1] which has been another criticism in theory despite their superior performance in practice [112]. Our analyses show that models based on large vocabularies find more separable anchor objects, learning stable latent clusters

[1] Each cluster has at least one anchor object which dedicates exclusively to that cluster, nothing else.

without much sensitivity to sample noise. Overall, we complete a robust and scalable pipeline: that efficiently performs quality posterior inference for unsupervised learning from co-occurrence information within time and space complexity linear in N.

The major contributions in this chapter are:

- We introduce two efficient rectifications, ENN and PALM, that compress quadratic and noisy co-occurrence information on the fly with a linear space rectified representation.
- We develop a low-rank algorithm (LAW) for anchor-word-based topic modeling that works directly on the compressed rectified representations and provides near-exact performance.
- We propose a robust and scalable pipeline, LR-JSMF, that learns topic models with a small number of passes directly over the data. This new pipeline scales to large vocabularies that were previously intractable for spectral inference, and offers a ~100× speedup over previous methods for general data sets.

5.2 Background and Related Work

Our new algorithms build on the the Joint-Stochastic Matrix Factorization (JSMF) framework [111], which we now describe. Let $H \in \mathbb{R}^{N \times M}$ be the object-example matrix whose m-th column vector h_m counts the occurrences of each of the N objects in the vocabulary in example m. We denote the total number of objects in example m by n_m. Given a user-specified number of clusters K, we seek to learn an object-cluster matrix $B \in \mathbb{R}^{N \times K}$ where B_{ik} is the conditional probability of observing object i given latent cluster k. Instead of learning B directly from the sparse and noisy observations H, JSMF

begins with constructing the joint-stochastic co-occurrence $C \in \mathbb{R}^{N \times N}$ by

$$C = \hat{H}\hat{H}^T - \hat{H}_{\text{diag}}, \quad \hat{h}_m = \frac{h_m}{\sqrt{n_m(n_m - 1)M}}, \quad \hat{H}_{\text{diag}} = \text{diag}\left(\sum_{m=1}^{M} \frac{h_m}{n_m(n_m - 1)M}\right). \quad (5.1)$$

Then the original Anchor Word algorithm decomposes C into BAB^T by Algorithm 5.1, where $A \in \mathbb{R}^{K \times K}$ is the cluster correlation matrix, whose entry A_{kl} captures the joint probability between two latent clusters k and l.[2] In the limit, using data generated according to the correct probabilistic models, A must agree with the second-moment of the cluster proportions, which is given as a Bayesian prior in the models.

Algorithm 5.1: Anchor Word algorithm (AW)

Input: Object co-occurrence $C \in \mathbb{R}^{N \times N}$
The number of clusters K
Output: Anchor objects $S = \{s_1, ..., s_K\}$
Latent clusters $B \in \mathbb{R}^{N \times K}$
Cluster correlations $A \in \mathbb{R}^{K \times K}$
begin
L_1-normalize the rows of C to form $\overline{C}$.
Find S via the column pivoted QR on $\overline{C}^T$.
Find $\breve{B}$ with $\breve{B}_{ki} = p(\text{cluster } k \mid \text{object } i)$ by solving N simplex-constrained least squares in parallel to minimize $\left\|\overline{C} - \breve{B}^T \overline{C}_{S*}\right\|_F$.
Recover B from $\breve{B}$ by the Bayes' rule.
Recover A by $B_{S*}^{-1} C_{SS} B_{S*}^{-1}$.
end

As with other spectral algorithms for latent variable models [89, 3], the decomposition as described so far may fail to learn high-quality clusters due to *model-data mismatch* [107]. Under the probabilistic model assumed to generate the data, the expected value of the co-occurrence should not only be normalized to sum to one ($\mathcal{NOR}$) and be entry-wise non-negative ($\mathcal{NN}$), but it should also be positive semi-definite with rank equal to the number of clusters K ($\mathcal{PSD}_K$) [111]. However, the empirical C from real

[2] C is proven to be a by far more robust estimator than H in [6]. But actual construction of C in [7] is slightly misleading without dividing by M. We report the full equations here.

Algorithm 5.2: Rectified AW algorithm (RAW)

Input/Output: Same as Algorithm 5.1

begin

$C_0 \leftarrow C$

repeat *with* $t = 0, 1, 2, ...$

$(U, \Lambda_K) \leftarrow$ Truncated-Eig(C_t, K)

$\Lambda_K^+ \leftarrow \max(\Lambda_K, 0)$

$C^{\mathcal{PSD}_K} \leftarrow U\Lambda_K^+ U^T$

$C^{\mathcal{NOR}} \leftarrow C^{\mathcal{PSD}_K} + \frac{1-\sum_{ij} C_{ij}^{\mathcal{PSD}_K}}{N^2} ee^T$

$C^{\mathcal{NN}} \leftarrow \max(C^{\mathcal{NOR}}, 0)$

$C_{t+1} \leftarrow C^{\mathcal{NN}}$

until *converging to a certain* C^*

$(S, B, A) \leftarrow$ AW(C^*, K) (Algorithm 5.1)

end

data is often indefinite and full-rank due to sample noise [3] and the unbiased construction of C in Eq. (5.1), which penalizes all diagonal entries. The **Rectified Anchor Word (RAW)** algorithm has an additional rectification step that enforces that C should enjoy the expected structures before running the main algorithm. In [111], an Alternating Projection (AP) rectification as given in Algorithm 5.2 is used to overcome the gap between the underlying assumptions of our models and the actual data.

Rectification is also important for addressing the issue of *outlier bias*. Real data often exhibits rare objects that are only present in a few examples. The corresponding co-occurrence of these objects are inevitably sparse with large variance, but the greedy anchor selection favors choosing these outliers due to L_1 normalization of C. Previous work tried to bypass this problem by oversampling clusters by the number of outliers under some additional identifiability assumptions [69]. This approach is not always feasible, especially for a large vocabulary that introduces many low frequency objects. When synonyms and short documents cause undesirable sparsity to Latent Semantic

[3] Rectification still improves the quality of clusters on the synthetic data that is generated from our models.

Analysis [108], projection onto the leading eigensubspace blurs sparse co-occurrences. Similarly, $\mathcal{PSD}_K$-projection turns out to significantly reduce outlier bias, and the remaining projections are useful for maintaining the probabilistic structures of C, which then allow users to recover B and A in Algorithm 5.1.

Handling a *large vocabulary* is another major challenge for spectral methods. Even if we limit our focus only to second-order models, the space complexity of RAW is already $O(N^2)$, growing rapidly with increasing vocabulary. We are unable to exploit the high sparsity of C as a single iteration of the AP-rectification makes C significantly denser. The three projections in AP-rectification and the rest of the anchor word algorithm in Algorithm 5.1 have time complexities of $O(N^2K)$, $O(N^2)$, $O(N^2)$ and $O(N^2K)$, respectively, and so pose a difficulty when scaling to a large vocabulary size N. On the other hand, the *separability assumption* is crucial for the second-order models, and while there has been a line of research that tries to relax this assumption [17, 94], it has been formally shown that most topic models are indeed separable if their vocabulary sizes are sufficiently larger than the number of clusters [48], again emphasizing the urgency of an approach with better time and space scaling in the vocabulary size.

5.3 Low-rank Rectification and Compression

The rectified co-occurrence C^* in Section 5.2 must be of rank K and positive semidefinite, hinting at an opportunity to represent it in terms of an outer product YY^T for some $Y \in \mathbb{R}^{N\times K}$. One idea for achieving this structure is to use a low-rank representation $C_t = Y_t Y_t^T$ throughout the rectification in Algorithm 5.2. Another way to obtain this structure is to directly minimize $\left\|C - YY^T\right\|_F$ with the necessary constraints. In this section, we propose two new algorithms to simultaneously compress and rectify the input

by representing $C \in \mathbb{R}^{N \times N}$ by a low-rank outer product YY^T.

5.3.1 Epsilon Non-Negative Rectification

The alternating projection iteration in Algorithm 5.2 produces low-rank semi-definite intermediate matrices in factored form at each iteration. By construction, the positive semi-definite projection ($\mathcal{PSD}_K$) and the normalization projection ($\mathcal{NOR}$) produce positive semi-definite matrices of rank K and $K + 1$, respectively. Unfortunately, the final projection to enforce elementwise non-negativity ($\mathcal{NN}$) destroys this low rank structure. However, the $\mathcal{NN}$ projection only makes significant changes to a few elements; that is, the output of the $\mathcal{NN}$ projection at step t is nearly rank $K + 1$ plus a sparse correction E_t. The Epsilon Non-Negative Rectification algorithm (Algorithm 5.3) has the same structure as Algorithm 5.2, but with the key difference that it returns a sparse-plus-low-rank representation of the $\mathcal{NN}$ projection rather than materializing a dense representation. Matrix-vector products with this sparse-plus-low-rank representation require $O(NK + \text{nnz}(E_t))$ time, and $O(K)$ such matrix-vector products can be used in a Lanczos eigensolver to compute the truncated eigendecomposition at the start of the next iteration.

Maintaining a sparse correction matrix E_t at each step lets the ENN approach avoid the storage overheads of the original alternating projection algorithm. To overcome the quadratic time cost at each iteration, though, we need to avoid explicitly computing every element of the intermediate YY^T in the course of the $\mathcal{NN}$ projection. However, we can bound the magnitude of many elements of YY^T by the Cauchy-Schwartz inequality: $\left|C_{ij}\right| \leq \left\|y_i\right\|_2 \left\|y_j\right\|_2$ where y_i and y_j denote columns of Y^T. Let I denote the index set indices $\{i : \left\|y_i\right\|_2^2 > \epsilon\} \subseteq [N]$ for given ϵ; then every large entry of C belongs to either

Algorithm 5.3: ENN-rectification (ENN)

Input: Object co-occurrence $C \in \mathbb{R}^{N \times N}$
The number of clusters K
Output: Rectified compression $Y \in \mathbb{R}^{N \times K}$
begin
  $E \leftarrow 0 \in \mathbb{R}^{N \times N}$ (sparse format)
  $C^{op} : x \rightarrow Cx$ (Implicit operator)
  repeat *with* $t = 0, 1, 2, ...$
    $(U, \Lambda_K) \leftarrow$ Truncated-Eig(C^{op}, K)
    $\Lambda_K^+ \leftarrow \max(\Lambda_K, 0)$
    $Y \leftarrow U(\Lambda_K^+)^{1/2}$
    $E_{ij} \leftarrow \max(-Y_{i*}Y_{j*}^T, 0)$
    $r \leftarrow (1 - \left\|Y^T e\right\|_2^2 - \sum_{ij} E_{ij})/N^2$
    $C^{op} : x \rightarrow Y(Y^T x) + Ex + r(e^T x)e$
  until *E converges*
end

$Y_{I*}Y^T$ or $Y(Y^T)_{*I}$. As C is symmetric, checking the negative entries in $Y_{I*}Y^T$ is sufficient to find a symmetric correction E that guarantees $YY^T + E \geq -\epsilon$. We refer to this property as *Epsilon Non-Negativity*: ϵ balances the trade-off between the effect of leaving small negative entries versus increasing the size of I to look up. We limit $|I|$ to be $\bar{O}(K)$ based on the common sampling complexity of a suitable set of rows for a near-optimal rank-K approximation [4].

5.3.2 Proximal Alternating Linearized Minimization Rectification

To avoid small negative entries, we investigate another rectified compression algorithm that directly minimizes $\left\|C - YY^T\right\|_F$ subject to the stronger $\mathcal{NN}$-constraint $Y \geq 0$ and the usual $\mathcal{NOR}$-constraint $\left\|Y^T e\right\|_2 = 1$. Concretely, we try to

$$\text{minimize} \;\; J(X, Y) := \frac{1}{2}\left\|C - XY^T\right\|_F^2 + \frac{s}{2}\|X - Y\|_F^2 \quad \text{subject to} \;\; X \geq 0, Y \geq 0\,. \quad (5.2)$$

[4] This choice is standard in literature on low-rank approximation via column subset selection.

$\mathcal{PSD}_K$- and $\mathcal{NOR}$-constraints are implicitly satisfied by jointly minimizing the two terms in the objective function J, whereas $\mathcal{NN}$-constraint is explicit in the formulation. Thus we can apply the Proximal Alternating Linearized Minimization (PALM)[27] for learning Y given C; the relevant proximal operator is $\mathcal{NN}$ projection of Y, which takes $O(NK)$ at most.

Note that J is semi-algebraic (as it is a real polynomial) with two partial derivatives: $\nabla_X J = (XY^T - C)Y + s(X - Y)$ and $\nabla_Y J = (YX^T - C)X + s(Y - X)$. So the following lemma guarantees the convergence to a critical points, under the assumption that X, Y remain bounded during the iterations.

Lemma 5.1. *For any fixed Y, $\nabla_X J(X, Y)$ is globally Lipschitz continuous with the moduli $L_1(Y) = \left\|Y^T Y + sI_K\right\|_2$. So is $\nabla_Y J(X, Y)$ given any fixed X with $L_2(X) = \left\|X^T X + sI_K\right\|_2$.*

Proof.

$$\begin{aligned} & \left\|\nabla_X J(X, Y) - \nabla_X J(X', Y)\right\|_F \\ = & \left\|(Y^T Y + sI_K)(X - X')\right\|_F \\ \leq & \left\|Y^T Y + sI_k\right\|_2 \cdot \left\|X - X'\right\|_F \end{aligned}$$

The proof is symmetric for the other case with $L_2(X) = \left\|X^T X + sI_K\right\|_2$. □

Algorithm 5.4 shows the PALM-rectification with the adaptive control of the learning rates based on the tight 2-norm Lipschitz modulis defined in Lemma 5.1 at each step t. In our experiments, we chose $\gamma = 2.0$ and $s = 1e-4$ based on our empirical observations on multiple datasets.

Algorithm 5.4: PALM-rectification (PALM)

Input: Object co-occurrence $C \in \mathbb{R}^{N \times N}$
The number of clusters K
Output: Rectified compression $Y \in \mathbb{R}^{N \times K}$
begin
  $(V, D) \leftarrow$ Truncated-Eig(C, K)
  $(X_0, Y_0) \leftarrow (V\sqrt{D}, V\sqrt{D})$
  repeat
    $c_t \leftarrow \gamma L_1(Y_t)$
    $X'_{t+1} \leftarrow X_t - (1/c_t)\nabla_X J(X_t, Y_t)$
    $X_{t+1} \leftarrow \max(X'_{t+1}, 0)$
    $d_t \leftarrow \gamma L_2(X_{t+1})$
    $Y'_{t+1} \leftarrow Y_t - (1/d_t)\nabla_Y J(X_{t+1}, Y_t)$
    $Y_{t+1} \leftarrow \max(Y'_{t+1}, 0)$
  until *Y converges*
end

5.4 Low-rank Anchor Word Algorithm

The output for both methods in Section 5.3 is a compressed co-occurrence matrix $C = YY^T$. In this section, we present the **Low-rank Anchor Word algorithm (LAW)** that reduces the time complexity of finding anchor objects from $O(N^2K)$ to $O(NK^2)$ by taking advantage of this form. We note that LAW applies whenever C is in a low-rank representation, which does not have to be derived from our methods. Moreover, it is exact for non-negative C, but robust in practice when we allow small negative entries in C, as in the case with ENN.

The first step is to L_1-normalize the rows of C. Given $C \geq 0$, the L_1-norm of each row is simply the sum of all its entries, so we can calculate the row norms by $d = Y(Y^T e)$. To obtain the normalized C, we simply scale the rows of Y, and $\overline{C} = (\text{diag}(d)^{-1}Y)Y^T = \overline{Y}Y^T$. These steps cost $O(NK)$.

Next, we need to apply column pivoted QR to $\overline{C}^T$ in order to identify the pivots

Algorithm 5.5: Low-rank AW (LAW)

Input: Object co-occurrence $C = YY^T$

Output: Anchor objects $S = \{s_1, ..., s_K\}$
Latent clusters $B \in \mathbb{R}^{N \times K}$
Cluster correlations $A \in \mathbb{R}^{K \times K}$

begin

Calculate row sums $d = Y(Y^T e)$.

Normalize $\overline{Y} \leftarrow \text{diag}(d)^{-1} Y$.

Compute QR decomposition of $Y = QR$.

Form $X = \overline{Y} R^T$.

Select S using column pivoted QR on X^T.

Solve n simplex-constrained least square problems to minimize $\left\| X - \breve{B} X_{S*} \right\|_F$.

Recover B from $\breve{B}$ using Bayes' rule.

Recover $A = B_{S*}^{-1} Y_{S*} Y_{S*}^T B_{S*}^{-1}$.

end

Algorithm 5.6: Low-rank JSMF (LR-JSMF)

Input: Raw object-example $H \in \mathbb{R}^{N \times M}$

Output: Anchor objects $S = \{s_1, ..., s_K\}$
Latent clusters $B \in \mathbb{R}^{N \times K}$
Cluster correlations $A \in \mathbb{R}^{K \times K}$

begin

Get $\hat{H}, \hat{H}_{\text{diag}}$ from H by Eq. (5.1).

$C^{op} : x \to \hat{H}(\hat{H}^T x) - \hat{H}_{\text{diag}} x$

$(U, \Lambda_K) \leftarrow$ Randomized-Eig(C^{op}, K)

Initialize ENN with U, Λ_K.

$Y \leftarrow$ ENN-rectification

$(S, B, A) \leftarrow$LAW(Y) (Algorithm 5.5)

end

as our anchor objects S. By taking the QR decomposition $Y = QR$, $\overline{C}^T$ can be further transformed into $QR\overline{Y}^T$. Notice that $\overline{C}^T$ is an orthogonal embedding of $X^T = R\overline{Y}^T$ onto a higher-dimensional space, which preserves the column L_2-norms. Lemma 5.2 shows that column pivoted QR on $\overline{C}^T$ and on $R\overline{Y}^T$ are equivalent, which allows us to lower the computation cost from $O(N^2K)$ to $O(NK^2)$.

Lemma 5.2. *Let S be the set of pivots that have been selected by column pivoted QR on $\overline{C}^T = QX^T$. Given the QR decomposition, $\overline{X}^T_{S*} = PT$, then $\overline{C}^T_{S*} = (QP)T$ is the corresponding QR decomposition for the columns of $\overline{C}$. For any remaining row $i \in [N] \setminus S$,*

$$\left\|(I - PP^T)\overline{X}^T_{i*}\right\|_2 = \left\|\left(I - (QP)(QP)^T\right)\overline{C}^T_{i*}\right\|_2 . \tag{5.3}$$

Therefore, the next column pivot is identical for $\overline{C}^T$ and $\overline{X}^T$. By induction, column pivoted QR on $\overline{C}^T$ and $\overline{X}^T$ return the same pivots.

Proof. Because both Q and P have orthonormal columns,

$$(QP)^T(QP) = P^T(Q^TQ)P = P^TP = I . \tag{5.4}$$

Thus, QP and T forms the QR decomposition of $\overline{C}^T_{S*}$. The residual of a remaining column $i \in [N] \setminus S$ is $\left(I - (QP)(QP)^T\right)\overline{C}^T_{i*}$ and $(I - PP^T)\overline{X}^T_{i*}$ for $\overline{C}^T$ and $\overline{\boldsymbol{X}}^T$, respectively. Simplify the former gives us

$$\begin{aligned}
&\left(I - (QP)(QP)^T\right)\overline{C}^T_{i*} \\
&= \left(I - (QP)(QP)^T\right)Q\overline{X}^T_{i*} \\
&= Q\overline{X}^T_{i*} - QPP^TQ^TQ\overline{X}^T_{i*} \\
&= Q(I - PP^T)\overline{X}^T_{i*}
\end{aligned}$$

Finally,

$$\left\|\left(I - (QP)(QP)^T\right)\overline{C}^T_{i*}\right\|_2^2 = \overline{X}_{i*}(I - PP^T)Q^TQ(I - PP^T)\overline{X}^T_{i*} = \left\|(I - PP^T)\overline{X}^T_{i*}\right\|_2^2 \tag{5.5}$$

Because the next pivot is selected as the column whose residual has the largest L_2-norm, Eq. 5.5 indicates that the same pivot will be selected for $\overline{C}^T$ and $\overline{X}^T$. Inductively, the anchors S recovered by column pivoted QR on those matrices are equivalent. □

Following the recovery of S, AW solves N independent simplex-constrained least square problems $\left\|\overline{C}_{i*} - \check{B}_{i*}^T \overline{C}_{S*}\right\|_2$. Again we can leverage the L_2-norm preserving property,

$$\left\|\overline{C}_{i*} - \check{B}_{i*}^T \overline{C}_{S*}\right\|_2 = \left\|X_{i*}Q^T - \check{B}_{i*}^T X_{S*} Q^T\right\|_2 = \left\|X_{i*} - \check{B}_{i*}^T X_{S*}\right\|_2 \tag{5.6}$$

and reduce the dimension of the least-square problems from N to K, thereby the complexity from $O(N^2K)$ to $O(NK^2)$. The remaining part of the algorithm follows exactly as AW.

Low-rank Joint Stochastic Matrix Factorization (LR-JSMF) We complete our scalable framework of processing co-occurrence statistic by introducing a direct initialization method from the raw object-example data for ENN. This allows us to avoid creating and storing C, which is a burden of memory when N becomes sufficiently large. In Algorithm 5.3, C only appears in the initial truncated eigendecomposition, after which we maintain the compressed operator C^{op} independent of it. On the other hand, we just need the matrix-vector-multiplication by C for the iterative methods in initialization. Using the generative formula in Eq. (5.1), we are able to implicitly apply C to vectors as an outer-product plus diagonal operator, in terms of H, at $O(NMK)$ computation cost. To further reduce the number of times the operator is applied, we adopt the one-pass randomized eigendecomposition by Halko et al. [79]. This technique enables us to initialize with a single pass over the dataset, without concurrently storing the entire H in memory. A limitation is when the number of clusters is large and the gap between the K-th eigenvalue and the ones below is small, we will have to incor-

porate a few power iterations for refinement, as suggested by the original paper. This will result in a multi-pass method, but still far more efficient on large object size and parallelization-friendly.

5.5 Experimental Results

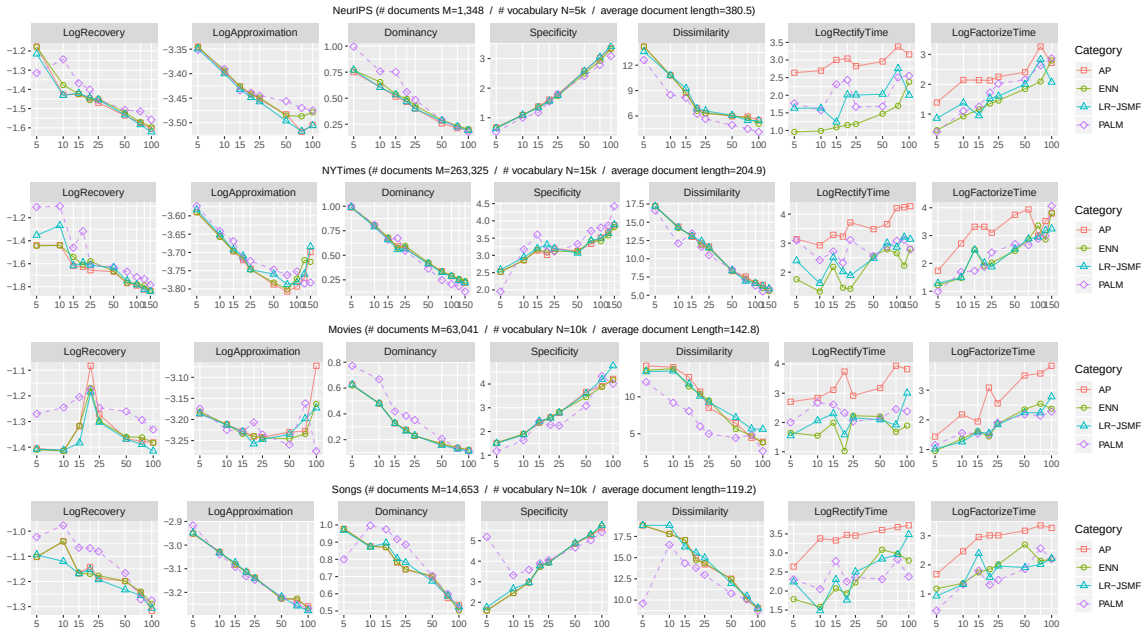


Figure 5.1: Experiment on four datasets. ENN and LR-JSMF mostly agree with AP, whereas PALM has slight inconsistency. The general information of each dataset is above the corresponding row. Recovery, approximation, and runtimes are in $\log_{10}$ scale. Note that ENN and LR-JSMF are almost two orders of magnitude faster than AP. The x-axis indicates the number of clusters K. Lower numbers in y-axis are better except Specificity and Dissimilarity.

A good factorization should be accurate, meaningful, and fast. In two series of experiments, we show that LR-JSMF maintains model quality while running in a fraction of the space and time needed for the original JSMF method. Previous methods have required truncation of the vocabulary even to run on consumer-grade computers. We show not only that we are able to handle increasingly large vocabularies without loss of speed, but that using larger vocabularies measurably improves model quality relative to

truncated vocabularies.

For the first series of experiments, we measure the accuracy of each rectification component as well as the entire pipeline of LR-JSMF. To produce a strong baseline, we begin with constructing the full co-occurrence C from each of our datasets H by Eq. (5.1), and produce the rectified C_{AP} by running Alternating Projection (AP) on C. Next we compress C into Y_{ENN} and Y_{PALM} by running ENN (50 iterations, $|I| = 10K + 1000$) and PALM (100 iterations, $s = 1e-4$). For testing our complete low-rank pipeline, we also construct (V, D) directly from the raw data H by the randomized eigendecomposition in Algorithm 5.6, learning the compressed statistics $Y_{LR-JSMF}$ again by running ENN initialized with $V\sqrt{D}$. Then we run the Anchor Word algorithm (AW) on C_{AP} and the Low-rank Anchor Word algorithm (LAW) on each of Y_{ENN}, Y_{PALM}, and $Y_{LR-JSMF}$.

The goal of rectification is to apply spectral inference to data that does not follow our modeling assumptions, so we evaluate on real data. In addition to two standard datasets from the UCI Machine Learning repository (NeurIPS papers and New York Times articles), we also use two non-textual datasets (Movies and Songs) previously used to demonstrate the performance of full algorithm with AP-rectification in [111]. Although our ultimate goal is to extend JSMF to large vocabularies, we use the same restricted vocabulary as [111] for a fair comparison in the first series of experiment.

Fig. 5.1 shows the overall performance of the learned topic clusters from these four datasets with increasing number of clusters K. Low **Recovery** error $\frac{1}{N}\sum_i \left\|\overline{C}_{i*} - \breve{B}_{i*}\overline{C}_{S*}\right\|_2$ implies that the learned anchor objects successfully reconstruct the co-occurrence space of the entire objects. Low **Approximation** error $\left\|C - BAB^T\right\|_2$ means that our factorization captures most of information given in the unbiased co-occurrence statistics. In real data, low **Dominancy** $\frac{1}{K}\sum_k A_{kk}$ implies that our models learn

more correlations between clusters. High **Specificity** $\frac{1}{K}\sum_k \mathrm{KL}(B_{*k}\|\sum_i C_{*i})$ indicates that the learned clusters are different enough from the corpus distribution, whereas high **Dissimilarity** counts the average number of objects in each cluster that do not occur among top 20 in other clusters, showing the interpretable difference across the learned clusters. We do not report the Cluster Coherence because it often measures deceptively [33]. The first five columns show that ENN and LR-JSMF learn approximately same clusters as JSMF with the full AP, showing no visible loss in accuracy across all settings. More importantly, the randomness we introduced into LR-JSMF results in a very low variance over a number of runs. This is important as the stability of spectral inference is a major advantage relative to MCMC or Variational Inference. Although PALM deviates a small amount from the other three methods in a few cases, it mostly achieves the same level of accuracy and follows the overall trend closely. In terms of runtimes, all of our methods have clear advantage over AP, gaining $1 \sim 2$ orders of magnitude speedup in most situations. Even for applications on relatively small vocabulary sizes, our algorithms shows a notable improvement in efficiency.

For the second series of experiments, we create eight corpora $\{H_N, H_{2N}, \cdots, H_{8N}\}$ for each dataset H by tailoring their vocabulary sizes as multiples of a base vocabulary of N objects. In this case we are not able to compare LR-JSMF to previous methods because we cannot store the full co-occurrence matrices for the larger vocabulary: these models would be impossible. Fig. 5.2 illustrates the overall performance of the learned small/medium/large size clusters with increasing vocabulary size N. Low **BaseRecovery** means that the anchor objects from the models with larger vocabulary better reconstruct the objects in our base vocabulary (H_N). High **AnchorQuality** indicates that the average rank of the anchor objects s_k in every other topic clusters than k is high, implying the anchor objects rarely contribute to other clusters than their own. High **Sparsity** $\left(\frac{1}{K}\sum_k \frac{\sqrt{N}-(\|b_k\|_1/\|b_k\|_2)}{\sqrt{N}-1}\right)$ [88] says that our topic clusters are more concentrated on specific

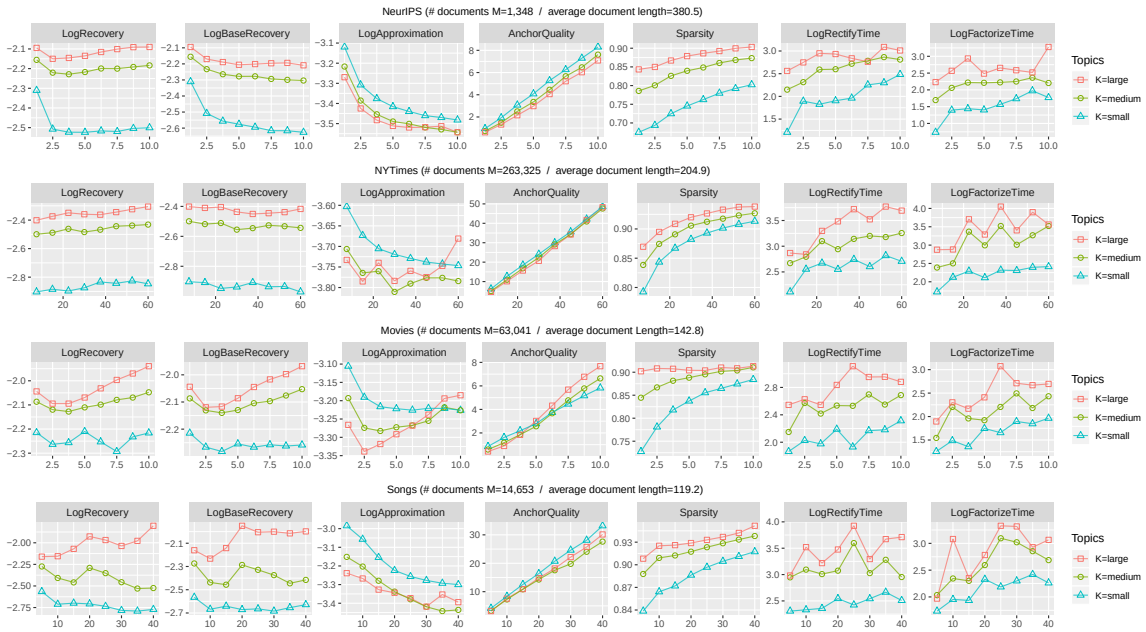


Figure 5.2: As we increase the vocabulary size for four collections, anchor quality and sparsity improve, but running time is stable. The x-axis indicates the vocabulary size N in thousands. Values above 15k will not fit in memory on standard hardware with previous algorithms.

objects.

We observe the quality of anchors increases with increasing vocabulary size, verifying that using larger vocabularies helps better satisfy the separability assumption. We also verify that a large vocabulary often better approximates the co-occurrence statistics and better reconstructs the co-occurrence space of the *base* vocabulary, but these patterns are not always consistent in non-textual datasets. In contrast, Sparsity consistently improves, increasing the interpretability of the learned clusters. Most excitingly, the running times of ENN and LAW show the scalability of our new rectification and low-rank algorithm, thereby demonstrating that LR-JSMF is an efficient and robust pipeline.

Finally, we have also inspected the qualitative behavior of the recovered clusters, as we increase the vocabulary size. The topic clusters become significantly more specific, while the clustering of objects is more conspicuous. Fig. 5.3 shows how using a

larger vocabulary size can lead to more distinguishable topics, especially as it allows us to make use of words that are relatively rare, but used in much narrower contexts. Going from left to right, we can observe that the set of topic words become more and more specific: for instance, the topic corresponding to the third row is slightly vague when observing just the left half of the row, while as we increase the vocabulary size beyond 5000, we gain access to highly topic-specific words such as hjb (Hamilton-Jacobi-Bellman equation) or pid (Proportional Integral Derivative), which signifies the row's pertinence to dynamical and control systems. The flip side of the figure shows that words we would normally consider as non-topical can often be assigned high contributions towards certain topics. The strong red shade on the bottom left indicates that words such as "equivalent" or "cambridge" are strongly connected to the machine learning literature.

5.6 Conclusion

Spectral algorithms provide an appealing alternative for identifying interpretable low-rank subspaces by simple factorizations of higher-order moments. But this simplicity is also a weakness: violations of modeling assumptions destroy performance unless they are handled through rectification, and the size of the moment matrices limits us to small vocabularies. In this chapter, we developed an efficient and scalable framework: Low-Rank Joint Stochastic Matrix Factorization. We provide theoretical advances in compressed matrix factorization, leading to high-quality low-rank non-negative approximations without quadratic blowup. The method provides orders of magnitude speedups for rectification even on small vocabularies. Perhaps most importantly, we can now apply reliable, high-quality factorizations of high-dimensional data sets on laptop-grade hardware, massively increasing the applicability and potential use of these algorithms.

Vocabulary Size						
1250	2500	3750	5000 (base)	6250	7500	8750
refractory	interspike	bursting	neuron	signalling	ipsp	tst
interconnection	marder	stomatogastric	circuit	meilijson	stg	tam
seen	abbott	konishi	synaptic	quiescent	substances	hyperpolarized
detail	acad	axonal	cell	ryckebusch	inactivation	memorized
transmission	pyloric	modulatory	layer	leech	depolarized	transposed
considered	bird	ionic	signal	silent	shapiro	tsividis
san	male	kanji	recognition	radical	sicl	subword
additional	henderson	phonemic	layer	npm	joe	phoneroes
amount	jackel	subsystem	hidden	demi	chinese	otherfilter
considered	recog	dtw	word	shikano	hanazawa	sdnn
developed	dictionary	strokes	speech	letterform	lexicon	perplexity
significant	ocr	gender	net	tebelskis	preprocessed	males
cambridge	discounted	tutor	control	hjb	jacobi	ovi
pendulum	bradtke	lqr	action	rein	forcement	pid
requires	discount	disturbances	dynamic	biped	viscosity	idm
con	eligibility	disturbance	optimal	trol	sel	umass
bellman	indirect	hamilton	reinforcement	handicapped	bizzi	missile
plan	amherst	smdp	controller	gullapalli	swinging	queueing
directional	transparent	luminance	cell	unoriented	aftereffect	moc
seen	adelson	ruderman	field	heeger	blast	ori
dark	geniculate	andersen	visual	thalamus	mae	taube
neuroscience	amacrine	bergen	motion	directionally	knierim	muller
soc	deg	selectively	direction	hayashi	mexican	swindale
supported	mcnaughton	lond	image	werblin	specimen	skagg
equivalent	fix	solvable	gaussian	birmingham	boxplot	pbr
computing	satisfying	distribu	noise	kolmogorov	dependences	owi
cambridge	royal	barber	approximation	gmm	cb2	danish
considered	opper	parametrized	hidden	winther	trigonometric	ylz
simply	leibler	const	bound	imation	colt	diabetes
detail	treatment	eter	matrix	statist	minimizer	devroye

Figure 5.3: Losses or gains in topic words depending on the vocabulary size. Each row represents a topic from the NeurIPS dataset, with the top 6 topical words shown in the middle column. The red and green cells denote topic words that are lost or gained by shifting the vocabulary size from the default size 5000, respectively. The intensities of the colors indicate the words' contributions towards the specific topic.

CHAPTER 6

WEIGHTED K-MEANS FOR ELECTRONIC STRUCTURE CALCULATION

LIST OF SYMBOLS

C	Pairwise products of orbitals sampled at interpolation points	Page 88
$K(r, r')$	(Screened) Coulomb operator	Page 85
N	Number of orbitals	Page 84
N_μ	Number of interpolation points	Page 85
N_e	Number of electrons in the system	Page 84
Z	Pairwise products of discretized orbitals	Page 88
Θ	Matrix with interpolating vectors as columns	Page 88
$\hat{r}_\mu$	Interpolation points	Page 85
ψ_j	Single particle orbitals	Page 84
$\rho(r)$	Weight (electron density) function	Page 91
φ_i	Single particle orbitals	Page 84
$\zeta_\mu(r)$	Interpolating vectors	Page 85
c	Rank parameter for centroidal Voronoi tessellation	Page 98
C_μ	Clusters of Voronoi tessellation	Page 91
c_μ	Centroids of Voronoi tessellation	Page 91
r	Spatial vector in $\mathbb{R}^3$	Page 84

The recently developed interpolative separable density fitting (ISDF) decomposition is a powerful way for compressing the redundant information in the set of orbital pairs, and has been used to accelerate quantum chemistry calculations in a number of contexts. The key ingredient of the ISDF decomposition is to select a set of non-uniform grid points, so that the values of the orbital pairs evaluated at such grid points can be used to accurately interpolate those evaluated at all grid points. The set of non-uniform grid points, called the interpolation points, can be automatically selected by a QR factorization with column pivoting (QRCP) procedure. This is the computationally most expensive step in the construction of the ISDF decomposition. In this chapter, we propose a new approach to find the interpolation points based on the centroidal Voronoi tessellation (CVT) method, which offers a much less expensive alternative to the QRCP procedure when ISDF is used in the context of hybrid functional electronic structure calculations. The CVT method only uses information from the electron density, and can be efficiently implemented using a K-Means algorithm. We find that this new method achieves comparable accuracy to the ISDF-QRCP method, at a cost that is negligible in the overall hybrid functional calculations. For instance, for a system containing 1000 silicon atoms simulated using the HSE06 hybrid functional on 2000 computational cores, the cost of QRCP-based method for finding the interpolation points costs 38.1 s, while the CVT procedure only takes 0.7 s. We also find that the ISDF-CVT method also enhances the smoothness of the potential energy surface in the context of *ab initio* molecular dynamics (AIMD) simulations with hybrid functionals.

6.1 Introduction

Orbital pairs of the form $\{\varphi_i(r)\psi_j(r)\}_{i,j=1}^{N}$, where φ_i, ψ_j are single particle orbitals, appear ubiquitously in quantum chemistry. A few examples include the Fock exchange

operator, the MP2 amplitude, and the polarizability operator [171, 130]. When N is proportional to the number of electrons N_e in the system, the total number of orbital pairs is $N^2 \sim O(N_e^2)$. On the other hand, the number of degrees of freedom needed to resolve all orbital pairs on a dense grid is only $O(N_e)$. Hence as N_e becomes large, the set of all orbital pairs contains apparent redundant information. In order to compress the redundant information and to design more efficient numerical algorithms, many algorithms in the past few decades have been developed. Pseudospectral decomposition [140, 160], Cholesky decomposition [20, 104, 5, 129], density fitting (DF) or resolution of identity (RI) [159, 181], and tensor hypercontraction (THC) [149, 150] are only a few examples towards this goal. When the single particle orbitals φ_i, ψ_j are already localized functions, "local methods" or "linear scaling methods" [71, 30, 77, 143] can be applied to construct such decomposition with cost that scales linearly with respect to N_e. Otherwise, the storage cost of the matrix to represent all orbital pairs on a grid is already $O(N_e^3)$, and the computational cost of compressing the orbital pairs is then typically $O(N_e^4)$.

Recently, Lu and Ying [122] developed a new decomposition called the *interpolative separable density fitting* (ISDF) , which takes the following form

$$\varphi_i(r)\psi_j(r) \approx \sum_{\mu=1}^{N_\mu} \zeta_\mu(r) \left(\varphi_i(\hat{r}_\mu)\psi_j(\hat{r}_\mu)\right) . \tag{6.1}$$

For a given r, if we view $[\psi_i(r)\psi_j(r)]$ as a row of the matrix $\{\psi_i\psi_j\}$ discretized on a dense grid, then the ISDF decomposition states that all such matrix rows can be approximately expanded using a linear combination of matrix rows with respect to a selected set of *interpolation points* $\{\hat{r}_\mu\}_{\mu=1}^{N_\mu}$. The coefficients of such linear combination, or *interpolating vectors*, are denoted by $\{\zeta_\mu(r)\}_{\mu=1}^{N_\mu}$. Here N_μ can be interpreted as the numerical rank of the ISDF decomposition. Compared to the standard density fitting method, the three-tensor $\left\{\varphi_i(\hat{r}_\mu)\psi_j(\hat{r}_\mu)\right\}$ with three indices i, j, μ takes a separable form. This reduces the

storage cost of the decomposed tensor from $O(N_e^3)$ to $O(N_e^2)$, and the computational cost from $O(N_e^4)$ to $O(N_e^3)$. Note that if the interpolation points $\{\hat{r}_\mu\}_{\mu=1}^{N_\mu}$ are chosen to be on a uniform grid, then the ISDF decomposition reduces to the pseudospectral decomposition, where $N_\mu \sim O(N_e)$ but with a large preconstant. For instance, the pseudospectral decomposition can be highly inefficient for molecular systems, where the grid points in the vacuum contribute nearly negligibly to the orbital pairs. On the other hand, by selecting the interpolation points carefully, e.g. through a randomized *QR factorization with column pivoting* (QRCP) procedure [72], the number of interpolation points can be significantly reduced. The QRCP based ISDF decomposition has been applied to accelerate a number of applications, at least in the context of pseudopotential approximation where the wavefunctions are smooth, including two-electron integral computation [122], correlation energy in the random phase approximation [121], density functional perturbation theory [118], and hybrid density functional calculations [92]. For example, when iterative solvers are used for hybrid density functional calculations, the Fock exchange operator V_X defined in terms of a set of orbitals $\{\varphi_i\}$ needs to be repeatedly applied to another set of Kohn-Sham orbitals $\{\varphi_j\}$

$$\left(V_X[\{\varphi_i\}]\psi_j\right)(r) = -\sum_{i=1}^{N_e} \varphi_i(r) \int K(r,r')\varphi_i(r')\psi_j(r')dr'. \tag{6.2}$$

where $K(r,r')$ is the kernel for the Coulomb or the screened Coulomb operator. The integration in Eq. 6.2 is often carried out by solving Poisson-like equations, using e.g. a fast Fourier transform (FFT) method, and the computational cost is $O(N_e^3)$ with a large preconstant. This is typically the most time consuming component in hybrid functional calculations, and can be accelerated by the ISDF decomposition for the orbital pairs $\{\varphi_i\psi_j\}$.

In Ref. [122], the interpolation points and the interpolation vectors are determined simultaneously through randomized QRCP applied to $\{\psi_i(\boldsymbol{r})\psi_j(\boldsymbol{r})\}$ directly. We recently

found that the randomized QRCP procedure has $O(N_e^3)$ complexity but with a relatively large preconstant, and may not be competitive enough when used repeatedly. In order to overcome such difficulty, we proposed a different approach in Ref. [92] that determines the two parts separately and reduces the computational cost. We use the relatively expensive randomized QRCP procedure to find the interpolation points in advance, and only recompute the interpolation vectors whenever $\{\psi_i(r)\psi_j(r)\}$ has been updated using an efficient least squares procedure that exploits the separable nature of the matrix to be approximated. As a result, we can significantly accelerate hybrid functional calculations using the ISDF decomposition in all but the first *self consistent field* (SCF) iteration.

In this work, we further remove the need of performing the QRCP decomposition completely and, hence, significantly reduce the computational cost. Note that an effective choice of the set of interpolation points should satisfy the following two conditions. (1) The distribution of the interpolation points should roughly follow the distribution of the electron density. In particular, there should be more points when the electron density is high, and less or even zero points if the electron density is very low. (2) The interpolation points should not be very close to each other. Otherwise, matrix rows represented by the interpolation points are nearly linearly dependent, and the matrix formed by the interpolation vectors will be highly ill-conditioned. The QRCP procedure satisfies both (1) and (2) simultaneously, and thus is an effective way for selecting the interpolation points. Here we demonstrate that (1) and (2) can also be satisfied through a much simpler centroidal Voronoi tessellation (CVT) procedure applied to a weight vector such as the electron density.

The Voronoi tessellation technique has been widely used in computer science [11], as well as scientific and engineering applications, e.g., image processing[55], pattern recognition [146], and numerical integration [19]. The concept of Voronoi tessellation

can be simply understood as follows. Given a discrete set of weighted points, the CVT procedure divides a domain into a number of regions, each consisting of a collection of points that are closest to its weighted centroid. Here we choose the electron density as the weight, and the centroids as the interpolation points. The centroids must be located where the electron density is significant, and hence satisfy the requirement (1). The centroids are also mutually separated from each other by a finite distance due to the nearest neighbor principle, and hence satisfy the requirement (2). Although detailed analysis of the error stemming from such a choice of interpolation points is very difficult for general nonlinear functions, we find that the CVT procedure approximately minimizes the residual of the ISDF decomposition in Eq. (6.1). In practice, the CVT procedure only applies to one vector (the electron density) instead of $O(N_e^2)$ vectors and hence is very efficient.

We apply the ISDF-CVT method to accelerate hybrid functional calculations in a planewave basis set. We perform such calculations for different systems with insulating (liquid water), semiconducting (bulk silicon), and metallic (disordered silicon aluminum alloy) characters, as well as ab initio molecular dynamics (AIMD) simulations. We find that the ISDF-CVT method achieves similar accuracy to that obtained from the ISDF-QRCP method, with significantly improved efficiency. For instance, for a bulk silicon system containing 1000 silicon atoms computed on 2000 computational cores with kinetic energy cutoff being 10 Ha, the QRCP procedure finds the interpolation points with 38.1 s, while the CVT procedure only takes 0.7 s. Since the solution of the CVT procedure is continuous with respect to changes in the electron density, we also find that the CVT procedure produces a smoother potential energy surface than that by the QRCP procedure in the context of AIMD simulations.

The remainder of the chapter is organized as follows. We briefly introduce the ISDF

decomposition in Section 6.2. In Section 6.3 we describe the ISDF-CVT procedure and its implementation for hybrid functional calculations. We present numerical results of the ISDF-CVT method in Section 6.4, and conclude in Section 6.5.

6.2 Interpolative Separable Density Fitting (ISDF) decomposition

In this section, we briefly introduce the ISDF decomposition [122] evaluated using the method developed in Ref. [92], which employs a separate treatment of the interpolation points and interpolation vectors.

First, assume the interpolation points $\{\hat{r}_\mu\}_{\mu=1}^{N_\mu}$ are known, then the interpolation vectors can be efficiently evaluated using a least squares method as follows. Using a linear algebra notation, Eq. (6.1) can be written as

$$Z \approx \Theta C\,, \tag{6.3}$$

where each column of Z is given by $Z_{ij}(r) = \varphi_i(r)\psi_j(r)$ sampled on a dense real space grids $\{r_i\}_{i=1}^{N_g}$, and $\Theta = [\zeta_1, \zeta_2, ..., \zeta_{N_\mu}]$ contains the interpolating vectors. Each column of C indexed by (i, j) is given by

$$\left[\varphi_i(\hat{r}_1)\psi_j(\hat{r}_1),\ \cdots,\ \varphi_i(\hat{r}_\mu)\psi_j(\hat{r}_\mu),\ \cdots,\ \varphi_i(\hat{r}_{N_\mu})\psi_j(\hat{r}_{N_\mu})\right]^T\ . \tag{6.4}$$

Eq. (6.3) is an overdetermined linear system with respect to the interpolation vectors Θ. The least squares approximation to the solution is given by

$$\Theta = ZC^T(CC^T)^{-1}. \tag{6.5}$$

It may appear that the matrix-matrix multiplications ZC^T and CC^T take $O(N_e^4)$ operations because the size of Z is $N_g \times N^2$ and the size of C is $N_\mu \times N^2$. However, both multiplications can be carried out with fewer operations due to the separable structure

of Z and C. The computational complexity for computing the interpolation vectors is $O(N_e^3)$, and numerical results indicate that the pre-constant is also much smaller than that involved in hybrid functional calculations [92]. Hence the interpolation vectors can be obtained efficiently using the least squares procedure.

The problem for finding a suitable set of interpolation points $\{\hat{r}_\mu\}_{\mu=1}^{N_\mu}$ can be formulated as the following linear algebra problem. Consider the discretized matrix Z of size $N_g \times N^2$, and find N_μ rows of Z so that the rest of the rows of Z can be approximated by the linear combination of the selected N_μ rows. This is called an interpolative decomposition [32], and a standard method to achieve such a decomposition is the *QR factorization with column pivoting* (QRCP) procedure [32] as

$$Z^T \Pi = QR. \tag{6.6}$$

Here Z^T is the transpose of Z, Q is an $N^2 \times N_g$ matrix that has orthonormal columns, R is an upper triangular matrix, and Π is a permutation matrix chosen so that the magnitude of the diagonal elements of R form an non-increasing sequence. The magnitude of each diagonal element R indicates how important the corresponding column of the permuted Z^T is, and whether the corresponding grid point should be chosen as an interpolation point. The QRCP factorization can be terminated when the $(N_\mu + 1)$-th diagonal element of R becomes less than a predetermined threshold. The leading N_μ columns of the permuted Z^T are considered to be linearly independent numerically. The corresponding grid points are chosen as the interpolation points. The indices for the chosen interpolation points $\{\hat{r}_\mu\}$ can be obtained from indices of the nonzero entries of the first N_μ columns of the permutation matrix Π.

The QRCP decomposition satisfies the requirements (1) and (2) discussed in Section 6.1. First, QRCP permutes matrix columns of Z^T with large norms to the front, and pushes matrix columns of Z^T with small norms to the back. Note that the square of the

vector 2-norm of the column of Z^T labeled by r is just

$$\sum_{i,j=1}^{N} \varphi_i^2(r)\varphi_j^2(r) = \left(\sum_{i=1}^{N} \varphi_i^2(r)\right)\left(\sum_{j=1}^{N} \psi_j^2(r)\right). \tag{6.7}$$

In the case when φ_i, ψ_j are the set of occupied orbitals, the norm of each column of Z^T is simply the electron density. Hence the interpolation points chosen by QRCP will occur where the electron density is significant. Second, once a column is selected, all other columns are immediately orthogonalized with respect to the chosen column. Hence nearly linearly dependent matrix columns will not be selected repeatedly. As a result, the interpolation points chosen by QRCP are well separated spatially.

It turns out that the direct application of the QRCP procedure (Eq. (6.6)) still requires $O(N_e^4)$ computational complexity. The key idea used in Ref. [122] to lower the cost is to randomly subsample columns of the matrix Z to form a smaller matrix $\widetilde{Z}$ of size $N_g \times \widetilde{N}_\mu$, where $\widetilde{N}_\mu$ is only slightly larger than N_μ. Applying the QRCP procedure to this subsampled matrix $\widetilde{Z}$ approximately yields the choice of interpolation points, but the computational complexity is reduced to $O(N_e^3)$. In the context of hybrid density functional calculations, the cost of the randomized QRCP method can be comparable to that of applying the exchange operator in the planewave basis set [92]. However, the ISDF decomposition can still significantly reduce the computational cost, since the interpolation points only need to be performed once for a fixed geometric configuration.

6.3 Centroidal Voronoi Tessellation based ISDF decomposition

In this section, we demonstrate that the interpolation points can also be selected from a Voronoi tessellation procedure. For a d-dimensional space, the Voronoi tessellation partitions a set of points $\{r_i\}_{i=1}^{N_g} \subset \mathbb{R}^d$ into a number of disjoint cells. The partition is based on the distance of each point to a finite set of points, called its generators. In our

context, let $\{\hat{r}_\mu\}_{\mu=1}^{N_\mu}$ denote such a set of generators, and the corresponding cell, C_μ, of a given generator $\hat{r}_\mu$ is defined as a cluster of points

$$C_\mu = \{r_i \mid \text{dist}(r_i\,,\,\hat{r}_\mu) < \text{dist}(r_i\,,\,\hat{r}_\nu) \text{ for all } \nu \neq \mu\}\,. \tag{6.8}$$

The distance can be chosen to be any metric, e.g. the L_2 distance as $\text{dist}(r\,,\,r') = \left\|r - r'\right\|_2$. In the case when the distances of a point r to $\hat{r}_\mu, \hat{r}_\nu$ are exactly the same, we may arbitrarily assign r to one of the clusters.

The *centroidal Voronoi tessellation* (CVT) is a specific type of Voronoi tessellation in which the generator $\hat{r}_\mu$ is chosen to be the centroid of its cell. Given a weight function $\rho(r)$ (such as the electron density), the centroid of a cluster C_μ is defined as

$$c(C_\mu) = \frac{\sum_{r_j \in C_\mu} r_j\, \rho(r_j)}{\sum_{r_j \in C_\mu} \rho(r_j)}\,. \tag{6.9}$$

Combined with the L_2-distance, CVT can be viewed as a minimization problem over both all possible partition of the cells and the centroids as [127]

$$\{C_\mu^*, c_\mu^*\} = \underset{\{C_\mu, \boldsymbol{c}_\mu\}}{\arg\min} \sum_{\mu=1}^{N_\mu} \sum_{r_k \in C_\mu} \rho(r_k) \left\|r_i - c_\mu\right\|^2\,, \tag{6.10}$$

and the interpolation points are then chosen to be the minimizers $\hat{r}_\mu = c_\mu(C_\mu^*) = c_\mu^*$. Following the discussion in Section 6.1, the electron density as the weight function (6.10) enforces that the interpolation points should locate at points where the electron density is significant and hence satisfies the requirement (1). Since the cells C_μ^* are disjoint, the centroids c_μ^* are also separated by a finite distance away from each other and hence satisfies the requirement (2). Because the ISDF decomposition is a highly nonlinear process, in general we cannot expect the choice of interpolation points from CVT decomposition to maximally reduce the error of the decomposition. Instead, we demonstrate that the choice of the interpolation points from CVT approximately minimizes the residual for the ISDF decomposition, and hence provides a heuristic solution to the problem of finding interpolation points.

Theorem 6.1. *When the set of electron orbitals $\{\varphi_i\}$ are Lipschitz continuous, CVT method approximately minimizes the residual error of the ISDF decomposition.*

Proof. For simplicity we assume the limiting case where $\varphi_i = \psi_i$, and hence each row of Z is $Z(r) = [\varphi_i(r)\varphi_j(r)]_{i,j=1}^{N}$.

Now suppose we cluster all matrix rows of Z into sub-collections $\{C_\mu\}_{\mu=1}^{N_\mu}$, and for each C_μ we choose a representative matrix row $Z(r_\mu)$. Then the error of the ISDF can be approximately characterized as

$$R = \sum_{\mu=1}^{N_\mu} \sum_{r_k \in C_\mu} \left\| Z(r_k) - \mathrm{Proj}_{\mathrm{Span}\{Z(r_\mu)\}} Z(r_k) \right\|^2, \tag{6.11}$$

where the projection is defined according to the L_2 inner product as

$$\mathrm{Proj}_{\mathrm{Span}\{Z(r_\mu)\}} Z(r_k) = \frac{Z(r_k) \cdot Z(r_\mu)}{Z(r_\mu) \cdot Z(r_\mu)} Z(r_\mu). \tag{6.12}$$

Let Φ be the $N_g \times N$ matrix with each row $\Phi(r) = [\varphi_i(r)]_{i=1}^{N}$, then the electron density $\rho(r)$ is equal to $\Phi(r) \cdot \Phi(r)$. Using the relation

$$Z(r_\mu) \cdot Z(r_\mu) = (\Phi(r_\mu) \cdot \Phi(r_\mu))^2 = \rho(r_\mu)^2, \tag{6.13}$$

we have

$$R = \sum_{\mu=1}^{N_\mu} \sum_{r_k \in C_\mu} \rho(r_k)^2 \left(1 - \frac{(\Phi(r_k) \cdot \Phi(r_\mu))^4}{\rho(r_k)^2 \rho(r_\mu)^2} \right) \tag{6.14}$$

$$= \sum_{\mu=1}^{N_\mu} \sum_{r_k \in C_\mu} \rho(r_k)^2 [1 - \cos^4(\theta(r_k, r_\mu))] \,. \tag{6.15}$$

Here $\theta(r_k, r_\mu)$ is the angle between the vectors $\Phi(r_k)$ and $\Phi(r_\mu)$. Since

$$\rho(r_k)[1 - \cos^4(\theta(r_k, r_\mu))] \le 2\Phi(r_k) \cdot \Phi(r_k) \sin^2(\theta(r_k, r_\mu)) \tag{6.16}$$

$$\le 2 \left\| \Phi(r_k) - \Phi(r_\mu) \right\|^2, \tag{6.17}$$

we have

$$R \leq 2 \sum_{\mu=1}^{N_\mu} \sum_{r_k \in C_\mu} \rho(r_k) \left\| \Phi(r_k) - \Phi(r_\mu) \right\|^2 \tag{6.18}$$

$$\approx 2 \sum_{\mu=1}^{N_\mu} \sum_{r_k \in C_\mu} \rho(r_k) \left\| \nabla_r \Phi(r_\mu) \right\|^2 \left\| r_k - r_\mu \right\|^2 . \tag{6.19}$$

If we bound the gradient of $\Phi(r)$ by its Lipschitz constant, or simply neglect the spatial inhomogeneity in the electron orbitals, we arrive at the minimization criterion for the centroidal Voronoi tessellation decomposition. □

Many algorithms have been developed to efficiently compute the Voronoi tessellation [134]. One most widely used method is the Llyod's algorithm [120], which in discrete case is equivalent to the K-Means algorithm [127]. The K-Means algorithm is an iterative method that greedily minimizes the objective by taking alternating steps between $\{C_\mu\}$ and $\{c_\mu\}$. In this work, we adopt a weighted version of the K-Means algorithm, which is demonstrated in Algorithm 6.1. Note that the K-Means algorithm can be straightforwardly parallelized. We distribute the grid points evenly at the beginning. The classification step is the most time consuming step, and can be locally computed for each group of grid points. After this step, the weighted sum and total weight of all clusters can be reduced from and broadcast to all processors for the next iteration.

In order to demonstrate the CVT procedure, we consider the weight function $\rho(r)$ given by the summation of four Gaussian functions in a 2D domain. The initial choice of centroids, given by 40 uniformly distributed random points, together with its associated Voronoi tessellation are plotted in Fig. 6.1 (a). Fig. 6.1 (b) demonstrates the converged centroids and the associated Voronoi tessellation using the weighted K-Means algorithm. We observe that the centroids concentrate on where the weight function is significant, and are well-separated.

Algorithm 6.1: Weighted K-Means Algorithm to Find Interpolation Points for Density Fitting

Input : Grid points $\{r_i\}_{i=1}^{N_g}$, Weight function $\rho(r)$, Initial centroids $\{c_\mu^{(0)}\}$
Output: Interpolation points $\{\hat{r}_\mu\}_{\mu=1}^{N_\mu}$
Set $t \leftarrow 0$
do
  Classification step:
  for $i = 1$ **to** N_g
    Assign point r_i to the cluster $C_\mu^{(t)}$ **if** $c_\mu^{(t)}$ is the closest centroid to r_i
  end

  Update step:
  for $\mu = 1$ **to** N_μ
    $c_\mu^{(t+1)} \leftarrow \sum_{r_j \in C_\mu^{(t)}} r_j \, \rho(r_j) / \sum_{r_j \in C_\mu^{(t)}} \rho(r_j)$
  end
  Set $t \leftarrow t + 1$
while $\{c_\mu^{(t)}\}$ not converged and maximum steps not reached
for $\mu = 1$ **to** N_μ
  Set $\hat{r}_\mu \leftarrow c_\mu^{(t)}$
end

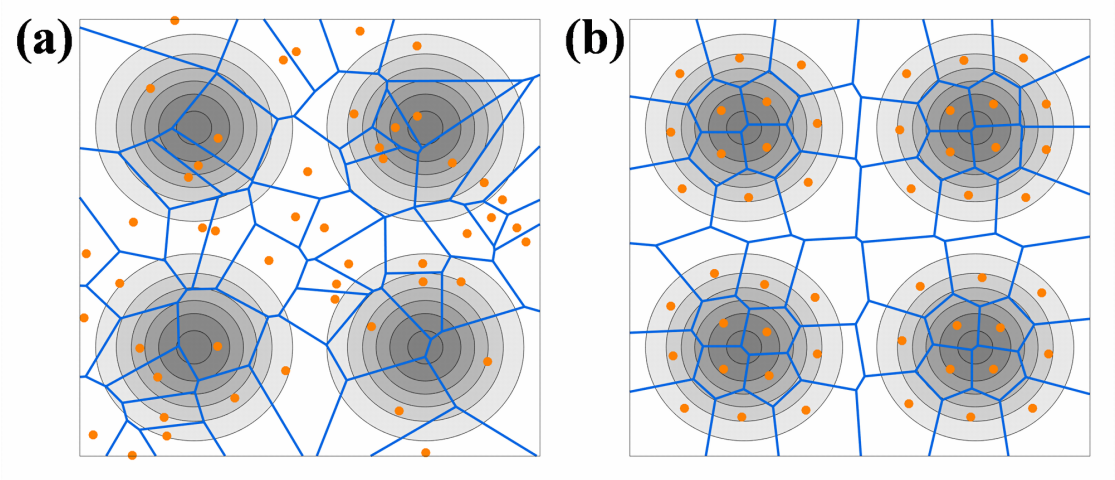


Figure 6.1: Schematic illustration of the CVT procedure in a 2D domain, including (a) initial random choice of centroids and Voronoi tessellation and centroidal Voronoi tessellation generated by the weighted K-Means algorithm. The weight function is given by the linear superposition of 4 Gaussian functions.

We also show how the interpolation points are placed and moved in real chemical systems, i.e., the ammonia-borane (BH_3NH_3) decomposition reaction process. Fig. 6.2 (a) shows the electron density of the molecule at the compressed, equilibrium, and dissociated configurations, respectively, according to the energy landscape in Fig. 6.2 (c). We plot the interpolation points found by the weighted K-Means algorithm in Fig. 6.2 (b). At the compressed configuration, all the interpolation points are distributed evenly around the molecule. As the bond length increases, some interpolation points are transferred from BH_3 to NH_3. Finally at the dissociated configuration, NH_3 has more interpolation points around the molecule, since there are more electrons in NH_3 than BH_3. Along the decomposition reaction process, both the transfer of the interpolation points and the potential energy landscape are smooth with respect to the change of the bond length.

6.4 Numerical Results

We demonstrate the accuracy and efficiency of the ISDF-CVT method for hybrid functional calculations by using the DGDFT (Discontinuous Galerkin Density Functional Theory) software package [117, 90, 91, 15, 193]. DGDFT is a massively parallel electronic structure software package designed for large scale DFT calculations involving up to tens of thousands of atoms. It includes a self-contained module called PWDFT for performing planewave based electronic structure calculations (mostly for benchmark and validation purposes). We implemented the ISDF-CVT method in PWDFT. We use the Message Passing Interface (MPI) to handle data communication. We use the Hartwigsen-Goedecker-Hutter (HGH) norm-conserving pseudopotential [82]. The atomic valence electron configuration is $1s^1$ for the H atom, $2s^22p^1$ for the B atom, $2s^22p^3$ for the N atom, $2s^22p^4$ for the O atom, $3s^23p^2$ for the Si atom in our DFT calcu-

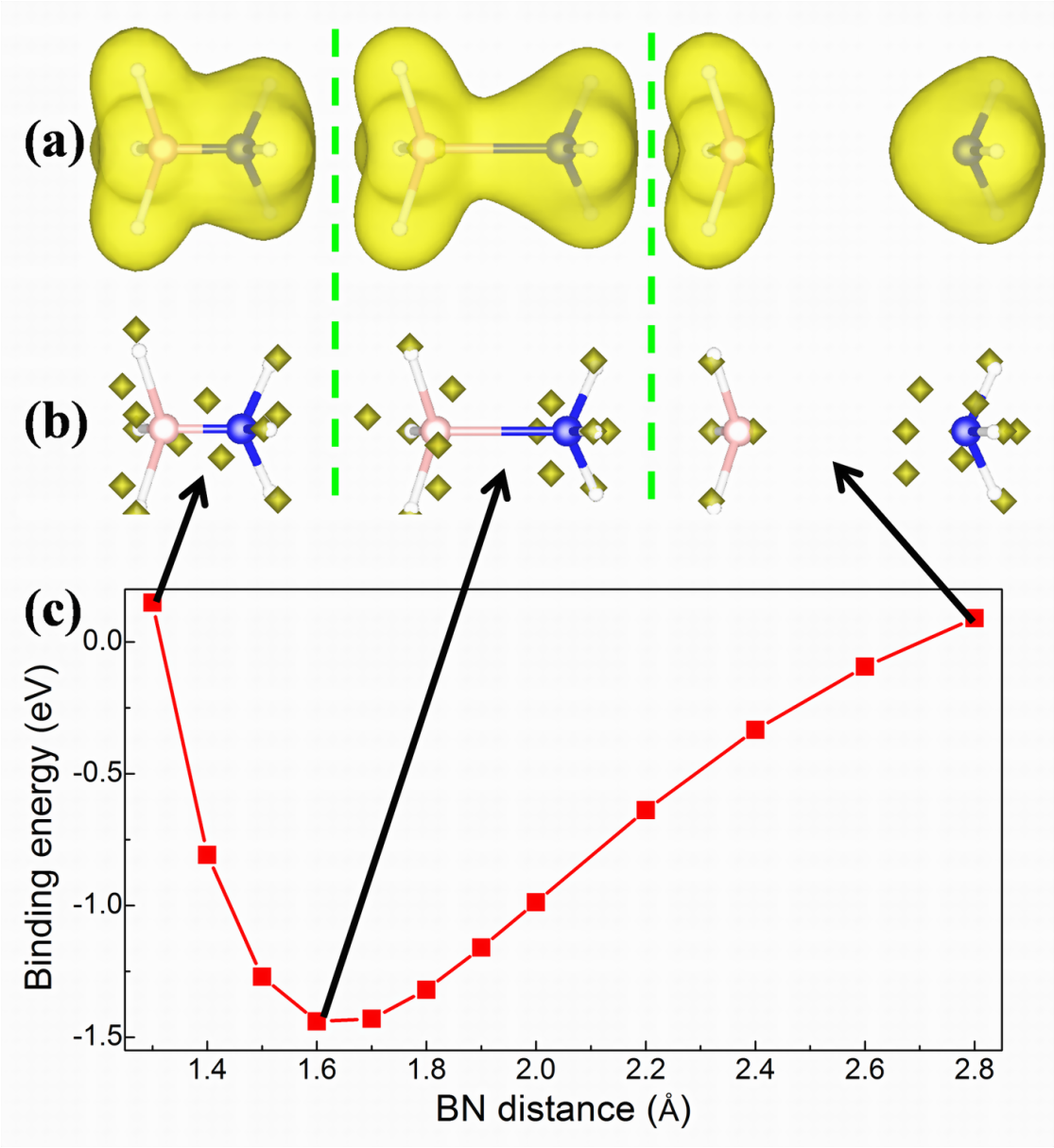


Figure 6.2: The decomposition reaction process of BH_3NH_3 computed with hybrid functional (HSE06) calculations by using the CVT procedure to select interpolation points, including (a) the electron density (yellow isosurfaces), (b) the interpolation points (yellow squares) $\{\hat{\boldsymbol{r}}_\mu\}_{\mu=1}^{N_\mu}$ ($N_\mu = 8$) selected from the real space grid points $\{\boldsymbol{r}_i\}_{i=1}^{N_g}$ ($N_g = 100^3$ and $E_{\text{cut}} = 60$ Ha) when the BN distance respectively is 1.3, 1.7 and 2.8 Å and (c) the binding energy as a function of BN distance for BH_3NH_3 in a 10 Å × 10 Å × 10 Å box. The white, pink and blue pink balls denote hydrogen, boron and nitrogen atoms, respectively.

lations, respectively. All calculations use the HSE06 functional [85], carried out on the Edison systems at the National Energy Research Scientific Computing Center (NERSC). Each node consists of two Intel "Ivy Bridge" processors with 24 cores in total and 64 gigabyte (GB) of memory. Our implementation only uses MPI. The number of cores is equal to the number of MPI ranks used in the simulation.

In this section, we demonstrate the performance of the ISDF-CVT method for accelerating hybrid functional calculations by using three types of systems [93]. They consist of bulk silicon systems (Si_{64}, Si_{216}, and Si_{1000}), a bulk water system with 64 molecules ($(H_2O)_{64}$), and a disordered silicon aluminum alloy system ($Al_{176}Si_{24}$). Bulk silicon systems (Si_{64}, Si_{216} and Si_{1000}) and bulk water system ($(H_2O)_{64}$) are semiconducting with a relatively large energy gap $E_{\text{gap}} > 1.0$ eV, and the $Al_{176}Si_{24}$ system is metallic with a small energy gap $E_{\text{gap}} < 0.1$ eV. All systems are closed shell systems, and the number of occupied bands is $N_{\text{band}} = N_e/2$, where N_e is the number of valence electrons. In order to compute the energy gap in the systems, we also include two unoccupied bands in all calculations.

6.4.1 Accuracy: Si_{216} and $Al_{176}Si_{24}$

We demonstrate the accuracy of the CVT-based ISDF decomposition in the hybrid functional calculation for semiconducting Si_{216} and metallic $Al_{176}Si_{24}$ systems, respectively. Although there is no general theoretical guarantee for the convergence of the K-Means algorithm and the convergence can depend sensitively on the initialization [8, 9], we find that, in the current context, initialization to have little impact on the final accuracy of the approximation. Hence we use random initialization for the K-Means algorithm. In all calculations, the *adaptively compressed exchange* (ACE) technique is used to accelerate

hybrid functional calculations without loss of accuracy [119]. The results obtained in this work are labeled as ACE-ISDF (CVT), which are compared against those obtained from the previous work based on the QRCP decomposition [92] labeled as ACE-ISDF (QRCP). In both cases, we introduce a rank parameter c to control the trade-off between efficiency and accuracy, by setting the number of interpolation points $N_\mu = cN_e$. We measure the error using the valence band maximum (VBM) energy level, the conduction band minimum (CBM) energy level, the energy gap, the Hartree-Fock exchange energy, the total energy, and the atomic forces, respectively. We remark that, in ISDF-CVT and ISDF-QRCP, the atomic force is computed directly using the Hellmann-Feynman formula, thereby neglects the Pulay force contribution from the change of the interpolation points. On the other hand, there is no Pulay contribution in the ACE formulation, and the Hellmann-Feynman force F_I^{ACE} can be used as the reference solution. The last three quantities are defined as:

$$\begin{aligned}
\Delta E_{\mathrm{HF}} &= \left| E_{\mathrm{HF}}^{\text{ACE-ISDF (CVT)}} - E_{\mathrm{HF}}^{\mathrm{ACE}} \right| / N_A \,, \\
\Delta E &= \left| E^{\text{ACE-ISDF (CVT)}} - E^{\mathrm{ACE}} \right| / N_A \,, \\
\Delta F &= \max_I \left\| F_I^{\text{ACE-ISDF (CVT)}} - F_I^{\mathrm{ACE}} \right\| \,,
\end{aligned}$$

where N_A is the number of atoms and I is the atom index.

Table 6.1 shows that the accuracy of the ACE-ISDF (CVT) method can systematically improve as the rank parameter c increases. When the rank parameter is large enough ($\geq$20.0), the results from ACE-ISDF (CVT) are fully comparable (the energy error is below 10^{-6} Ha/atom and the force error is below 10^{-5} Ha/Bohr) to those obtained from the benchmark calculations. Furthermore, for a moderate choice of the rank parameter $c = 6.0$, the error of the energy per atom reaches below the chemical accuracy of 1 kcal/mol (1.6×10^{-3} Ha/atom), and the error of the force is around 10^{-3} Ha/Bohr. This is comparable to the accuracy obtained from ACE-ISDF (QRCP), and to, e.g., lin-

ear scaling methods for insulating systems with reasonable amount of truncation needed to achieve significant speedup [46]. In fact, when compared with ACE-ISDF (QRCP) in Fig. 6.3, we find that the CVT based ISDF decomposition achieves slightly higher accuracy, though there is no theoretical guarantee for this to hold in general. The last column of Table 6.1 shows the runtime of the K-Means algorithm. As c increases, the number of interpolation points as well as the number of cells increases proportionally. Hence we observe that the runtime of K-Means scales linearly with respect to c.

6.4.2 Efficiency: $\mathbf{Si}_{1000}$

We report the efficiency of the ISDF-CVT method by performing hybrid DFT calculations for a bulk silicon system with 1000 atoms ($N_{\text{band}} = 2000$) on 2000 computational cores as shown in Table 6.2, with respect to various choices of the kinetic energy cutoff (E_{cut}). With the number of interpolation points fixed at $N_\mu = 12000$, both QRCP and K-Means scales linearly with the number of grid points N_g. Yet the runtime of K-Means is around two orders of magnitude faster than QRCP. The determination of interpolation vectors, which consists of solving a least square problem, previously costs a fifth of the ISDF runtime but now becomes the dominating component in CVT-based ISDF decomposition. Notice that the ISDF method allows us to reduce the number of Poisson-like equations from $N_e^2 = 4 \times 10^6$ to $N_\mu = 12000$, which results in a significant speedup in terms of the cost of the FFT operations.

Table 6.1: Accuracy of ACE-ISDF Based Hybrid Functional Calculations (HSE06) Obtained by Using the CVT method To Select Interpolation Points, with Varying Rank Parameter c for Semiconducting Si_{216} and Metallic $Al_{176}Si_{24}$ Systems[α].

c	E_{VBM}	E_{CBM}	E_{gap}	ΔE_{HF}	ΔE	ΔF	T_{KMEANS}
ACE-ISDF: Semiconducting Si_{216} (N_{band} = 432)							
4.0	6.7467	8.3433	-1.5967	2.69E-03	3.08E-03	5.04E-03	0.228
5.0	6.6852	8.2231	-1.5379	9.46E-04	1.12E-03	2.29E-03	0.248
6.0	6.6640	8.1522	-1.4882	3.76E-04	4.62E-04	1.05E-03	0.301
7.0	6.6550	8.1163	-1.4613	1.55E-04	1.98E-04	6.49E-04	0.312
8.0	6.6510	8.1030	-1.4520	7.33E-05	9.55E-05	3.07E-04	0.349
9.0	6.6490	8.0980	-1.4490	3.60E-05	4.96E-05	2.30E-04	0.398
10.0	6.6479	8.0959	-1.4480	1.78E-05	2.64E-05	1.30E-04	0.477
12.0	6.6472	8.0945	-1.4473	4.46E-06	8.91E-06	8.37E-05	0.530
16.0	6.6469	8.0937	-1.4468	1.51E-07	1.41E-06	3.20E-05	0.773
20.0	6.6468	8.0935	-1.4467	4.06E-07	3.33E-07	1.20E-05	0.830
24.0	6.6468	8.0935	-1.4467	2.99E-07	1.06E-07	5.18E-06	0.931
ACE	6.6468	8.0934	-1.4466	0.00E+00	0.00E+00	0.00E+00	-
ACE-ISDF: Metallic $Al_{176}Si_{24}$ (N_{band} = 312)							
4.0	7.9258	8.0335	-0.1076	3.80E-03	4.03E-03	8.01E-03	0.430
5.0	7.8537	7.9596	-0.1059	1.60E-03	1.69E-03	3.18E-03	0.535
6.0	7.8071	7.9127	-0.1056	6.07E-04	6.39E-04	1.48E-03	0.611
7.0	7.7843	7.8860	-0.1017	2.07E-04	2.17E-04	1.03E-03	0.731
8.0	7.7749	7.8749	-0.1000	7.43E-05	7.77E-05	4.40E-04	0.948
9.0	7.7718	7.8710	-0.0992	3.02E-05	3.20E-05	1.98E-04	0.947
10.0	7.7709	7.8697	-0.0989	1.48E-05	1.60E-05	1.80E-04	1.096
12.0	7.7703	7.8690	-0.0987	4.64E-06	5.60E-06	8.51E-05	1.305
16.0	7.7702	7.8688	-0.0986	6.35E-07	1.41E-06	3.24E-05	1.646
20.0	7.7701	7.8687	-0.0986	1.70E-08	5.30E-07	1.91E-05	2.037
ACE	7.7701	7.8687	-0.0986	0.00E+00	0.00E+00	0.00E+00	-

[α] The unit for VBM (E_{VBM}), CBM (E_{CBM}) and the energy gap E_{gap} is eV. The unit for the error in the Hartree-Fock exchange energy ΔE_{HF} and the total energy ΔE is Ha/atom, and the unit for the error in atomic forces ΔF is Ha/Bohr. We use the results from the ACE-enabled hybrid functional calculations as the reference. The last column shows the time (in seconds) for K-Means with different c values, with 434 cores for Si_{216} and 314 cores for $Al_{176}Si_{24}$ on Edison.

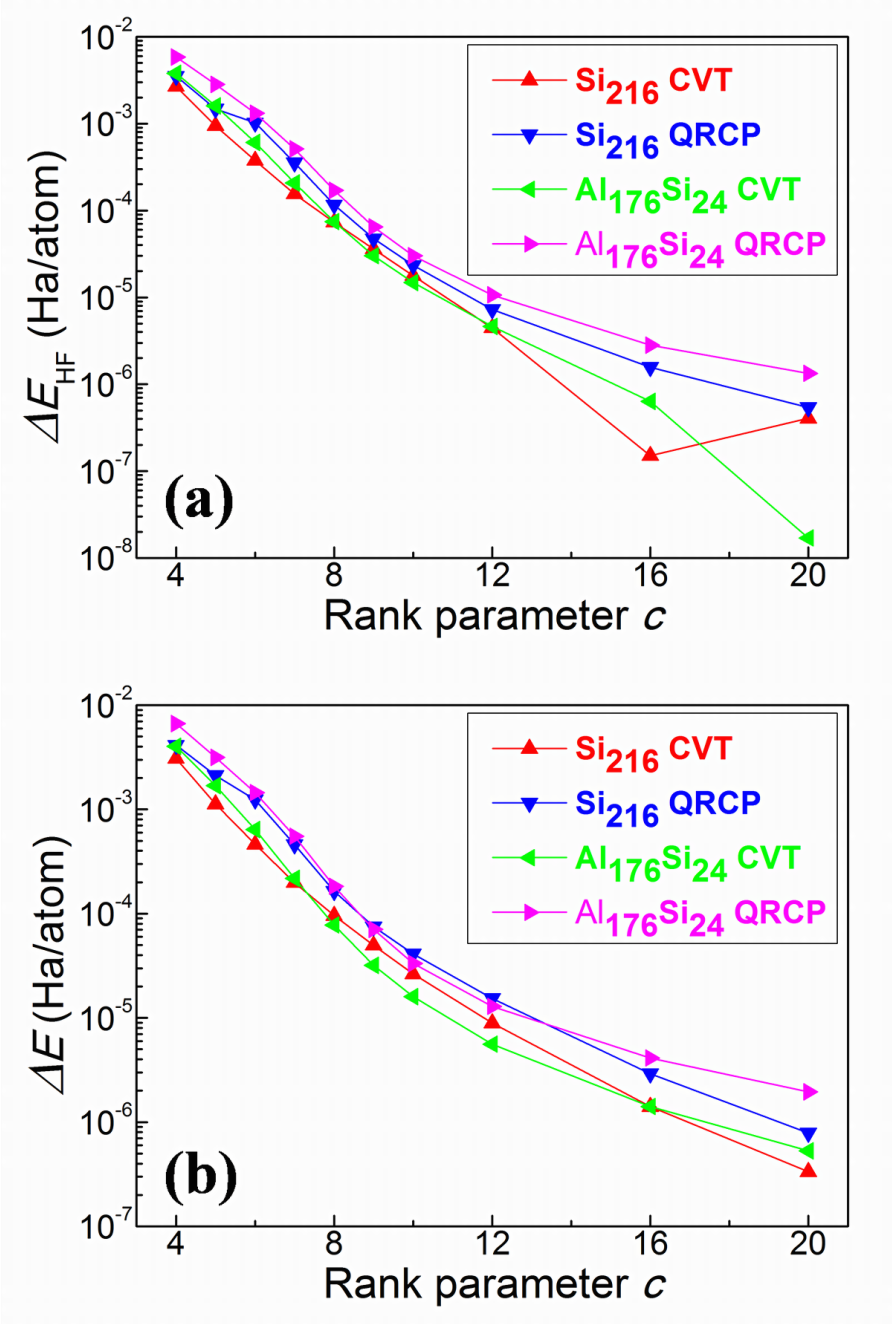


Figure 6.3: The accuracy of ACE-ISDF based hybrid functional calculations (HSE06) obtained by using the CVT and QRCP procedures to select the interpolation points, with varying rank parameter c from 4 to 20 for Si_{216} and $Al_{176}Si_{24}$, including the error of (a) Hartree-Fock exchange energy ΔE_{HF} (Ha/atom) and (b) total energy ΔE (Ha/atom).

Table 6.2: Wall Clock Time (in seconds) Spent in the Components of the ACE-ISDF and ACE Enabled Hybrid DFT Calculations Related to the Exchange Operator, for Si_{1000} on 2002 Edison cores at Different E_{cut} Levels[α].

Si_{1000}		ACE-ISDF			ACE
E_{cut}	N_g	IP_{QRCP}	IP_{KMEANS}	IV (FFT)	FFT
10	74^3	38.06	0.70	12.48 (0.33)	85.15
20	104^3	126.39	1.24	36.48 (0.71)	143.54
30	128^3	240.87	2.03	68.50 (1.43)	268.88
40	148^3	434.16	3.26	108.18 (3.10)	783.27

[α] Interpolation points are selected via either the QRCP or CVT procedure with the same rank parameter $c = 6.0$. N_g is the number of grid points in real space.

6.4.3 Ab Initio Molecular Dynamics: Si_{64} and $(H_2O)_{64}$

In this section, we demonstrate the accuracy of the ACE-ISDF (CVT) method in the context of AIMD simulations for a bulk silicon system Si_{64} under the NVE ensemble [67], and a liquid water system $(H_2O)_{64}$ under the NVT ensemble [67], respectively. For the Si_{64} system, the initial MD structure (initial temperature $T = 300$ K) is optimized by hybrid DFT calculations, and we perform the simulation ($E_{cut} = 20$ Ha) for 1.0 ps with a MD time step of 1.0 fs. For the $(H_2O)_{64}$ system, we perform the simulation ($E_{cut} = 60$ Ha) for 2.0 ps with a MD time step of 0.5 fs to sample the radial distribution function after equilibrating the system starting from a prepared initial guess [49]. In this case, the Van der Waals (VdW) interaction is modeled at the level of the DFT-D2 method [76]. We use a single level Nose-Hoover thermostat [145, 87] at $T = 295$ K, and the choice of mass of the Nose-Hoover thermostat is 85000 au.

In the AIMD simulation, the interpolation points need to be recomputed for each atomic configuration. At the initial MD step, although the initialization strategy does not impact the accuracy of the physical observable, it can affect the convergence rate of the K-Means algorithm. We measure the convergence in terms of the fraction of points

that switch clusters during two consecutive iterations. Fig. 6.4 (a) shows the convergence of the K-Means algorithm with interpolation points initially chosen from a random distribution and from the QRCP solution, respectively. We find that the K-Means algorithm spends around half the number of iterations to wait for 0.1% of the points to settle on the respective clusters. However, these points often belong to the boundary of the clusters and have little effect on the positions of the centroids (interpolation points). Therefore, we decide to terminate K-Means algorithm whenever the fraction of points that switch clusters falls below the 0.1% threshold. It is evident that QRCP initialization leads to faster convergence than random sampling. However, in the AIMD simulation, a very good initial guess of the interpolation points can be simply obtained from those from the previous MD step. Fig. 6.4 (b) shows that the number of K-Means iterations in the MD simulation can be very small, which demonstrates the effectiveness of this initialization strategy.

Fig. 6.5 (a) shows that both the CVT-based and QRCP-based ISDF decomposition lead to controlled energy drift, defined as $E_{\text{drift}}(t) = (E_{\text{tot}}(t) - E_{\text{tot}}(0))/E_{\text{tot}}(0)$. In the NVE simulation on bulk silicon system Si_{64}, the energy drift per atom is 6.6×10^{-5}, 7.5×10^{-5} and 2.5×10^{-5} Ha/ps respectively for the ISDF-CVT, ISDF-QRCP, and the conventional nested two-level SCF iteration procedure, indicating that ISDF is a promising method for reducing the cost of hybrid functional calculations with controllable loss of accuracy. Fig. 6.5 (b) shows the total potential energy obtained by the three methods along the MD trajectory, and the difference among the three methods is more noticeable. This is due to the fact that ISDF decomposition is a low rank decomposition for the pair product of orbitals, which leads to error in the Fock exchange energy and hence the total potential energy. Nonetheless, we find that such difference mainly results in a shift of the potential energy surface along the MD trajectory, and hence has little affect on physical observables defined via relative potential energy differences. Furthermore, the

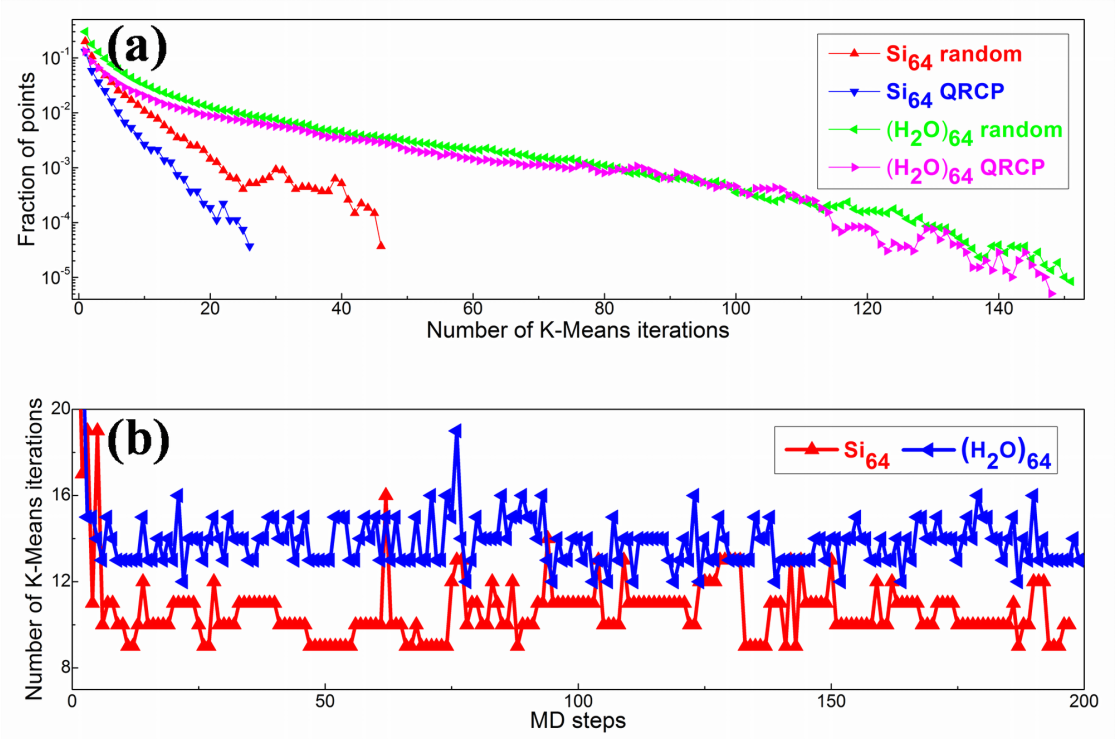


Figure 6.4: Comparison of the ISDF-CVT method by using either random or QRCP initialization for hybrid DFT AIMD simulations on bulk silicon system Si_{64} and liquid water system $(H_2O)_{64}$, including (a) the fraction of points what switch cluster in each K-Means iteration and (b) the number of K-Means iterations during each MD step.

CVT method yields a potential energy trajectory that is much smoother compared to that obtained from QRCP. This is because the interpolation points obtained from CVT are driven by the electron density, which varies smoothly along the MD trajectory. Such properties do not hold for the QRCP method. This means that the CVT method can be more effective when a smooth potential energy surface is desirable, such as in the case of geometry optimization. The absolute error of the potential energy from the CVT method is coincidentally smaller than that from QRCP, but again we are not aware of any reason for this behavior to hold in general.

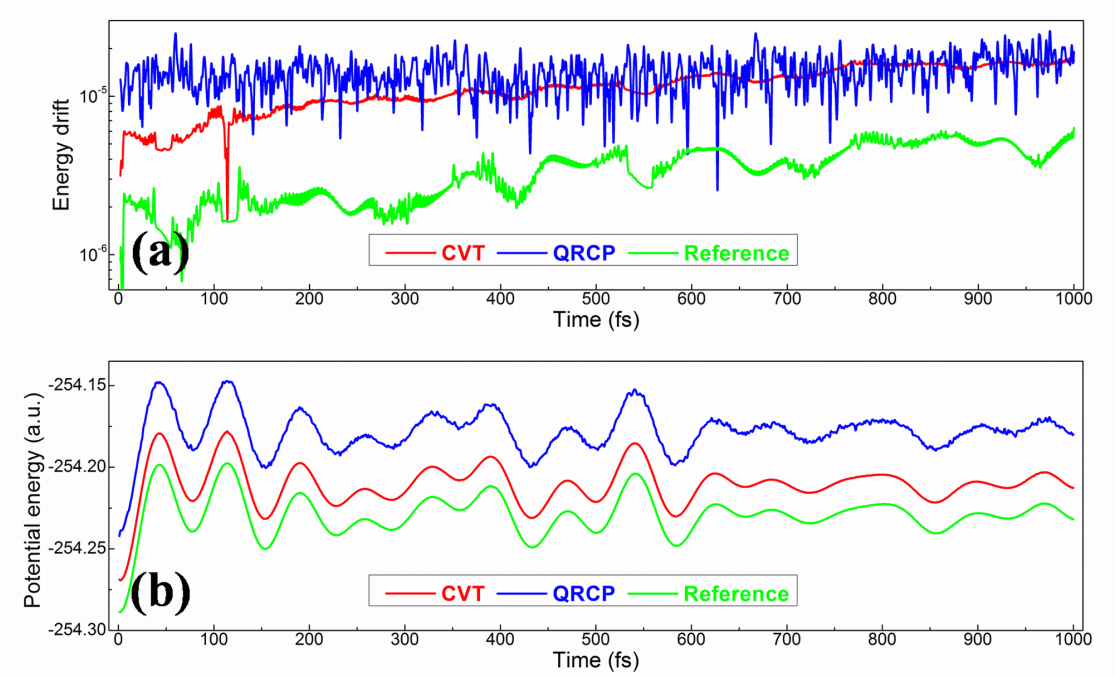


Figure 6.5: Comparison of hybrid HSE06 DFT AIMD simulations by using the ISDF-CVT and ISDF-QRCP methods as well as exact nested two-level SCF iteration procedure as the reference on the bulk silicon Si_{64}, including (a) relatively energy drift and (b) potential energy during MD steps.

We also apply the ACE-ISDF (CVT) and ACE-ISDF (QRCP) methods for hybrid DFT AIMD simulations on liquid water system $(H_2O)_{64}$ under the NVT ensemble to sample the radial distribution function in Fig. 6.6. We find that the results from all three methods agree very well, and our result is in quantitative agreement with previous hybrid functional calculations [49], which uses a different exchange-correlation functional (PBE0) and Van der Waals functional (TS-vdW) [172].

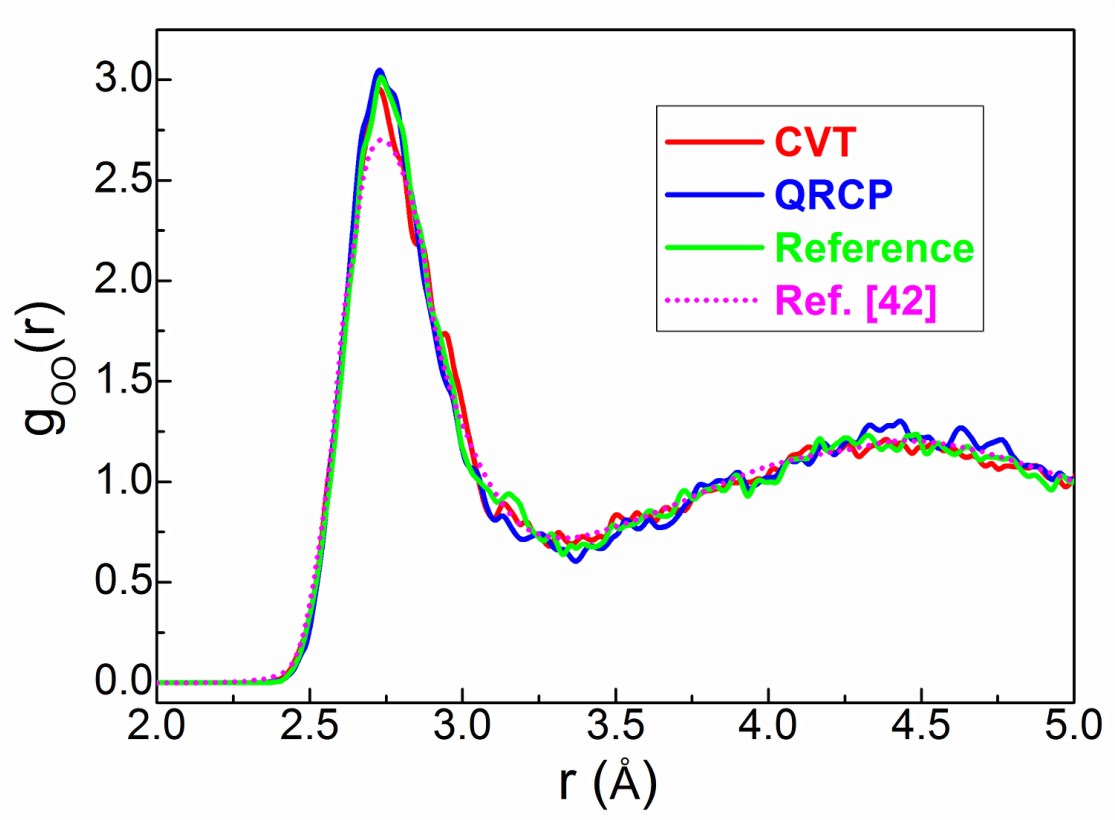


Figure 6.6: The oxygen-oxygen radial distribution functions $g_{\mathrm{OO}}(r)$ of liquid water system $(H_2O)_{64}$ at $T = 295$ K obtained from hybrid HSE06 + DFT-D2 AIMD simulations with the ISDF-CVT and ISDF-QRCP methods, exact nested two-level SCF iteration procedure (as the reference) as well as previous hybrid PBE0 + TS-vdW calculation [49].

6.5 Conclusion

In this work, we demonstrate that the interpolative separable density fitting decomposition (ISDF) can be efficiently performed through a separated treatment of interpolation points and interpolation vectors. We find that the centroidal Voronoi tessellation method (CVT) provides an effective choice of interpolation points using only the electron density as the input information. The resulting interpolation points are by design inhomogeneous in the real space, concentrated at regions where the electron density is significant, and are well separated from each other. These are all key ingredients for obtaining a low

rank decomposition that is accurate and a well conditioned set of interpolation vectors. We demonstrate that the CVT-based ISDF decomposition can be an effective strategy for reducing the cost hybrid functional calculations for large systems. The CVT-based method achieves similar accuracy when compared with that obtained from QRCP, with significantly improved efficiency. Since the solution of the CVT method depends continuously with respect to the electron density, we also find that the CVT method produces a smoother potential energy surface than that by the QRCP method in the context of ab initio molecular dynamics simulation. Our analysis indicates that it might be possible to further improve the quality of the interpolation points by taking into account the gradient information in the weight vector. We also expect that the CVT-based strategy can also be useful in other contexts where the ISDF decomposition is applicable, such as ground state calculations with rung-5 exchange-correlation functionals, and excited state calculations. These will be explored in the future work.

Acknowledgments. This work was partly supported by the National Science Foundation under grant No. DMS-1652330, the DOE under grant No. DE-SC0017867, the DOE CAMERA project (L. L.), and by the DOE Scientific Discovery through Advanced Computing (SciDAC) program (K. D., W. H. and L. L.). The authors thank the National Energy Research Scientific Computing (NERSC) center and the Berkeley Research Computing (BRC) program at the University of California, Berkeley for making computational resources available. We thank Anil Damle and Robert Saye for useful discussions.

CHAPTER 7

CONCLUSION

The rapid growth of data brings brand new opportunities and challenges to researchers across many fields, e.g., mathematics, computer science, statistics, scientific computing, and machine learning. There is more interest than ever in scalable numerical algorithms that take advantage of the abundant data. Randomized numerical linear algebra provides a powerful set of tools for handling large-scale matrix data. In this dissertation, we have presented a collection of randomized NLA algorithms for overcoming the computational obstacles in our problems. The applications we work with come from diverse areas, including network science, Gaussian process regression, natural language processing, and quantum chemistry. In all these cases, randomized NLA has led to efficient and robust algorithms, enabling us to extract valuable information from massive datasets. As the ongoing trend of large data persists, we expect randomized NLA to play a more and more important role in the development of practical algorithms. This dissertation has also left many interesting open questions, which we are excited to explore in the future.

In Chapter 3 we developed stochastic approximation methods for the spectral density of large real-world graphs. We were able to link the spectral characteristics of a graph to its structure properties. The spectral fingerprint of networks offers a new family of features that can be utilized in learning tasks. There are many directions available for future work. In this chapter, we mainly focused on normalized graph adjacency/Laplacian because of its popularity in graph partitioning. However, there exist many other graph matrices, each of which represents a unique aspect. Inspecting different types of graph matrices should provide new insights on graph structures. In addition, we are looking to extend the error analysis of our methods. From a practical perspective, we have

yet demonstrated the full power of spectral information in graph-related applications. In particular, we hope to incorporate the quantities we compute into graph clustering, graph classification, and node role discovery.

In Chapter 4 we proposed multiple methods to efficiently estimate the log determinant of kernel matrices, thus scaling Gaussian process regression to handle massive datasets. Our methods are highly flexible as they build upon the fast matrix-vector multiplication from existing kernel approximations. We showcased this generality by extending our approach to include derivative information. Together with dimensionality reduction through an active subspace method, we achieved efficient Bayesian optimization in high-dimensional space. Our algorithms have been incorporated into the GPyTorch library, a Python package for scalable Gaussian process regression. In the future, we look forward to applying our methods in other scenarios involving GPs.

In Chapter 5 we created a new scalable and robust framework for spectral inference of topic models. Through this pipeline, we are able to simultaneous compress and rectify large matrices of co-occurrence statistics, which then can be used for efficient inference of the underlying structure. Rectification, as a key ingredient in this work, helps correct the mismatch between proposed models and noisy data. The noise level in data varies significantly across applications, some of which are known for having low signal-noise ratio. Therefore, we hope to apply our scalable rectification to those applications, which will enhance the robustness of the existing methods. Moreover, the ϵ-nonnegativity is a rather heuristic approach toward our problem. In the future, we hope to find a better way of detecting negative entries in the outer product form XX^T.

In Chapter 6 we used centroid Voronoi tessellation, implemented as weighted K-means algorithm, to accelerate electronic structure calculation. As a replacement for the deterministic counterpart, our method brings tremendous speed-up without any loss

of accuracy in practice. Nonetheless, the remaining computation is still very time-consuming. Electronic structure calculation, along with other quantum chemistry problems, are among the most popular tasks on supercomputers. Therefore, it is very exciting to seek more opportunities where randomized NLA can improve the efficiency of those computation while maintaining high accuracy. On the other hand, there are many more numerical methods in this field that, like the kernel polynomial method, can be adapted to new applications. We hope to further explore these in the future.

BIBLIOGRAPHY

[1] Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *Journal of machine learning research*, 9 (Sep):1981–2014, 2008.

[2] A. Anandkumar, D. Hsu, and S. Kakade. A method of moments for mixture models and hidden markov models. In *COLT*, 2012.

[3] Anima Anandkumar, Dean P. Foster, Daniel Hsu, Sham Kakade, and Yi-Kai Liu. A spectral algorithm for latent Dirichlet allocation. In *NIPS*, 2012.

[4] Animashree Anandkumar, Daniel Hsu, and Sham M Kakade. A method of moments for mixture models and hidden markov models. In *Conference on Learning Theory*, pages 33–1, 2012.

[5] F. Aquilante, T. B. Pedersen, and R. Lindh. Low-cost evaluation of the exchange Fock matrix from Cholesky and density fitting representations of the electron repulsion integrals. *J. Chem. Phys.*, 126:194106, 2007.

[6] S. Arora, R. Ge, and A. Moitra. Learning topic models – going beyond SVD. In *FOCS*, 2012.

[7] Sanjeev Arora, Rong Ge, Yonatan Halpern, David Mimno, Ankur Moitra, David Sontag, Yichen Wu, and Michael Zhu. A practical algorithm for topic modeling with provable guarantees. In *ICML*, 2013.

[8] David Arthur and Sergei Vassilvitskii. How slow is the k-means method? In *Proceedings of the twenty-second annual symposium on Computational geometry*, pages 144–153. ACM, 2006.

[9] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[10] Subramanian Arumugam, Andreas Brandstädt, Takao Nishizeki, et al. *Handbook of graph theory, combinatorial optimization, and algorithms*. Chapman and Hall/CRC, 2016.

[11] Franz Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991.

[12] Haim Avron and Sivan Toledo. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *J. ACM*, 58(2):8:1–8:34, 2011. doi: 10.1145/1944345.1944349. URL `http://dx.doi.org/10.1145/1944345.1944349`.

[13] Haim Avron and Sivan Toledo. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *Journal of the ACM (JACM)*, 58(2):8, 2011.

[14] Zhaojun Bai, Mark Fahey, Gene H Golub, M Menon, and E Richter. Computing partial eigenvalue sums in electronic structure calculations. Technical report, Tech. Report SCCM-98-03, Stanford University, 1998.

[15] A. S. Banerjee, L. Lin, W. Hu, C. Yang, and J. E. Pask. Chebyshev polynomial filtered subspace iteration in the discontinuous galerkin method for large-scale electronic structure calculations. *J. Chem. Phys.*, 145:154101, 2016.

[16] Anirban Banerjee. *The spectrum of the graph Laplacian as a tool for analyzing structure and evolution of networks*. PhD thesis, 2008.

[17] Trapit Bansal, Chiranjib Bhattacharyya, and Ravindran Kannan. A provable SVD-based algorithm for learning topics in dominant admixture corpus. In *NIPS*, 2014.

[18] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[19] Axel D Becke. A multicenter numerical integration scheme for polyatomic molecules. *The Journal of chemical physics*, 88(4):2547–2553, 1988.

[20] N. H. F. Beebe and J. Linderberg. Simplifications in the generation and transformation of two-electron integrals in molecular calculations. *Int. J. Quantum Chem.*, 12:683, 1977.

[21] Costas Bekas, Effrosyni Kokiopoulou, and Yousef Saad. An estimator for the diagonal of a matrix. *Applied Numerical Mathematics*, 57(11-12):1214–1229, 2007.

[22] Costas Bekas, Efi Kokiopoulou, and Yousef Saad. An estimator for the diagonal of a matrix. *Applied Numerical Mathematics*, 57(11-12):1214–1229, November 2007. ISSN 0168-9274. doi: 10.1016/j.apnum.2007.01.003. URL `http://dx.doi.org/10.1016/j.apnum.2007.01.003`.

[23] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, pages 585–591, Cambridge, MA, USA, 2001. MIT Press.

[24] Einat Neumann Ben-Ari and David M Steinberg. Modeling data from computer experiments: an empirical comparison of kriging with MARS and projection pursuit regression. *Quality Engineering*, 19(4):327–338, 2007.

[25] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *JMLR*, 2003.

[26] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112 (518):859–877, 2017.

[27] Jérôme Bolte, Shoham Sabach, and Marc Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1-2):459–494, 2014.

[28] Christos Boutsidis, Michael W Mahoney, and Petros Drineas. An improved approximation algorithm for the column subset selection problem. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 968–977. SIAM, 2009.

[29] Christos Boutsidis, Petros Drineas, Prabhanjan Kambadur, Eugenia-Maria Kontopoulou, and Anastasios Zouzias. A randomized algorithm for approximating the log determinant of a symmetric positive definite matrix. *arXiv preprint arXiv:1503.00374*, 2015.

[30] D. R. Bowler and T. Miyazaki. O(n) methods in electronic structure calculations. *Rep. Prog. Phys.*, 75:036503, 2012. doi: doi:10.1088/0034-4885/75/3/036503.

[31] Martin Dietrich Buhmann. Radial basis functions. *Acta Numerica 2000*, 9:1–38, 2000.

[32] T. F. Chan and P. C. Hansen. Some applications of the rank revealing QR factorization. *SIAM J. Sci. Statist. Comput.*, 13:727–741, 1992.

[33] Jonathan Chang, Jordan Boyd-Graber, Chong Wang, Sean Gerrish, and David M. Blei. Reading tea leaves: How humans interpret topic models. In *NIPS*, 2009.

[34] Isaac Chavel. *Eigenvalues in Riemannian geometry*, volume 115. Academic press, 1984.

[35] Jeff Cheeger. A lower bound for the smallest eigenvalue of the laplacian. In *Proceedings of the Princeton conference in honor of Professor S. Bochner*, 1969.

[36] Fan Chung and Linyuan Lu. Connected components in random graphs with given expected degree sequences. *Annals of combinatorics*, 6(2):125–145, 2002.

[37] Fan Chung and Linyuan Lu. *Complex graphs and networks*. Number 107 in CBMS Regional Conference Series in Mathematics. American Mathematical Soc., 2006.

[38] Fan RK Chung and Fan Chung Graham. *Spectral graph theory*. Number 92. American Mathematical Soc., 1997.

[39] David Cohen-Steiner, Weihao Kong, Christian Sohler, and Gregory Valiant. Approximating the spectrum of a graph. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1263–1271. ACM, 2018.

[40] Puget Sound LiDAR Consortium. Mount Saint Helens LiDAR data. University of Washington, 2002.

[41] Paul G. Constantine. *Active subspaces: Emerging ideas for dimension reduction in parameter studies*. SIAM, 2015.

[42] Jane K Cullum and Ralph A Willoughby. *Lanczos algorithms for large symmetric eigenvalue computations: Vol. I: Theory*. SIAM, 2002.

[43] Kurt Cutajar, Michael Osborne, John Cunningham, and Maurizio Filippone. Preconditioning kernel matrices. pages 2529–2538, 2016.

[44] D. M. Cvetković, M. Doob, and H. Sachs. *Spectra of Graphs: Theory and Applications*. Wiley, third edition, 1998.

[45] Dragoš Cvetkovic, Slobodan Simic, and Peter Rowlinson. *An introduction to the theory of graph spectra*. Cambridge University Press, 2009.

[46] William Dawson and François Gygi. Performance and accuracy of recursive subspace bisection for hybrid dft calculations in inhomogeneous systems. *J. Chem. Theory Comput.*, 11(10):4655–4663, 2013. doi: 10.1021/acs.jctc.5b00826.

[47] Amit Deshpande and Luis Rademacher. Efficient volume sampling for row/column subset selection. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 329–338. IEEE, 2010.

[48] Weicong Ding, Prakash Ishwar, and Venkatesh Saligrama. Most large topic models are approximately separable. In *ITA, 2015*, pages 199–203. IEEE, 2015.

[49] R. A. DiStasio, B. Santra, Z. Li, X. Wu, and R. Car. The individual and collective effects of exact exchange and dispersion interactions on the ab initio structure of liquid water. *J. Chem. Phys.*, 141:084502, 2014.

[50] William E Donath and Alan J Hoffman. Lower bounds for the partitioning of graphs. In *Selected Papers Of Alan J Hoffman: With Commentary*, pages 437–442. World Scientific, 2003.

[51] Kun Dong, David Eriksson, Hannes Nickisch, David Bindel, and Andrew G Wilson. Scalable log determinants for gaussian process kernel learning. In *Advances in Neural Information Processing Systems*, pages 6327–6337, 2017.

[52] Kun Dong, Wei Hu, and Lin Lin. Interpolative separable density fitting through centroidal voronoi tessellation with applications to hybrid functional electronic

structure calculations. *Journal of chemical theory and computation*, 14(3):1311–1320, 2018.

[53] Kun Dong, Austin R Benson, and David Bindel. Network density of states. *arXiv preprint arXiv:1905.09758*, 2019.

[54] Petros Drineas and Michael W Mahoney. Randnla: randomized numerical linear algebra. *Communications of the ACM*, 59(6):80–90, 2016.

[55] Qiang Du, Max Gunzburger, Lili Ju, and Xiaoqiang Wang. Centroidal voronoi tessellation algorithms for image compression, segmentation, and multichannel restoration. *Journal of Mathematical Imaging and Vision*, 24(2):177–194, 2006.

[56] Francois Ducastelle and Françoise Cyrot-Lackmann. Moments developments and their application to the electronic charge distribution of d bands. *Journal of Physics and Chemistry of Solids*, 31(6):1295–1306, 1970.

[57] Nicole Eikmeier and David F Gleich. Revisiting power-law distributions in spectra of real world networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 817–826, 2017.

[58] David Eriksson. Scalable kernel methods and their use in black-box optimization. 2018.

[59] David Eriksson, Kun Dong, Eric Lee, David Bindel, and Andrew G Wilson. Scaling gaussian process regression with derivatives. In *Advances in Neural Information Processing Systems*, pages 6867–6877, 2018.

[60] Elena A. Erosheva. Bayesian estimation of the grade of membership model. *Bayesian Stat.*, 7, 01 2003.

[61] Illés J Farkas, Imre Derényi, Albert-László Barabási, and Tamas Vicsek. Spectra of "real-world" graphs: Beyond the semicircle law. *Physical Review E*, 64(2): 026704, 2001.

[62] Gregory E Fasshauer. *Meshfree approximation methods with MATLAB*, volume 6. World Scientific, 2007.

[63] M. Fiedler. Hankel and Loewner matrices. *Linear Algebra and Its Applications*, 58:75–95, 1984.

[64] Seth Flaxman, Andrew Wilson, Daniel Neill, Hannes Nickisch, and Alex Smola. Fast kronecker inference in gaussian processes with non-gaussian likelihoods. In *International Conference on Machine Learning*, pages 607–616, 2015.

[65] Alexander Forrester, Andy Keane, et al. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons, 2008.

[66] Jacob R. Gardner, Geoff Pleiss, Ruihan Wu, Kilian Q. Weinberger, and Andrew G. Wilson. Product kernel interpolation for scalable Gaussian processes. *arXiv preprint arXiv:1802.08903*, 2018.

[67] J Willard Gibbs. *Elementary principles in statistical mechanics*. Courier Corporation, 2014.

[68] Amparo Gil, Javier Segura, and Nico Temme. *Numerical Methods for Special Functions*. SIAM, 2007.

[69] Nicolas Gillis and Stephen A Vavasis. Fast and robust recursive algorithmsfor separable nonnegative matrix factorization. *IEEE transactions on pattern analysis and machine intelligence*, 36(4):698–714, 2014.

[70] David Gingras, Tom Lamarche, Jean-Luc Bedwani, and Érick Dupuis. Rough terrain reconstruction for rover motion planning. In *Proceedings of the Canadian Conference on Computer and Robot Vision (CRV)*, pages 191–198. IEEE, 2010.

[71] S. Goedecker. Linear scaling electronic structure methods. *Rev. Mod. Phys.*, 71: 1085–1123, 1999.

[72] G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins Univ. Press, Baltimore, fourth edition, 2013.

[73] Gene Golub and Gérard Meurant. *Matrices, Moments and Quadrature with Applications*. Princeton University Press, 2010.

[74] Gene H Golub and Gérard Meurant. Matrices, moments and quadrature ii; how to compute the norm of the error in iterative methods. *BIT Numerical Mathematics*, 37(3):687–705, 1997.

[75] Carolyn Gordon, David L. Webb, and Scott Wolpert. One cannot hear the shape of a drum. *Bull. Amer. Math. Soc.*, 27:134–138, 1992.

[76] S. Grimme. Semiempirical GGA-type density functional constructed with a long-range dispersion correction. *J. Comput. Chem.*, 27(15):1787–1799, 2006. doi: 10.1002/jcc.20495.

[77] M. Guidon, J. Hutter, and J. Vandevondele. Auxiliary density matrix methods for Hartree-Fock exchange calculations. *J. Chem. Theory Comput.*, 6:2348–2364, 2010.

[78] Raia Hadsell, J. Andrew Bagnell, Daniel F. Huber, and Martial Hebert. Space-carving kernels for accurate rough terrain estimation. *International Journal of Robotics Research*, 29:981–996, July 2010.

[79] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[80] Insu Han, Dmitry Malioutov, and Jinwoo Shin. Large-scale log-determinant computation through stochastic Chebyshev expansions. In *ICML*, pages 908–917, 2015.

[81] Helmut Harbrecht, Michael Peters, and Reinhold Schneider. On the low-rank approximation by the pivoted Cholesky decomposition. *Applied Numerical Mathematics*, 62(4):428–440, 2012.

[82] C. Hartwigsen, S. Goedecker, and J. Hutter. Relativistic separable dual-space gaussian pseudopotentials from H to Rn. *Phys. Rev. B*, 58:3641, 1998. doi: 10.1103/PhysRevB.58.3641.

[83] J Hensman, N Fusi, and N.D. Lawrence. Gaussian processes for big data. In *Uncertainty in Artificial Intelligence (UAI)*. AUAI Press, 2013.

[84] William Herlands, Andrew Wilson, Hannes Nickisch, Seth Flaxman, Daniel Neill, Wilbert Van Panhuis, and Eric Xing. Scalable Gaussian processes for characterizing multidimensional change surfaces. *Artificial Intelligence and Statistics*, 2016.

[85] J. Heyd, G. E. Scuseria, and M. Ernzerhof. Erratum: "hybrid functionals based on a screened coulomb potential" [J. Chem. Phys. 118, 8207 (2003)]. *J. Chem. Phys.*, 124:219906, 2006. doi: 10.1063/1.2204597.

[86] Nicholas J Higham. *Functions of matrices: theory and computation*. SIAM, 2008.

[87] W. G. Hoover. Canonical dynamics: Equilibrium phase-space distributions. *Phys. Rev. A*, 31:1695, 1985. doi: 10.1103/PhysRevA.31.1695.

[88] Patrik O. Hoyer. Non-negative matrix factorization with sparseness constraints. *JMLR*, 2004.

[89] Daniel Hsu, Sham M Kakade, and Tong Zhang. A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480, 2012.

[90] W. Hu, L. Lin, and C. Yang. DGDFT: A massively parallel method for large scale density functional theory calculations. *J. Chem. Phys.*, 143(12):124110, 2015. doi: 10.1063/1.4931732.

[91] W. Hu, L. Lin, and C. Yang. Edge reconstruction in armchair phosphorene nanoribbons revealed by discontinuous galerkin density functional theory. *Phys. Chem. Chem. Phys.*, 17(47):31397–31404, 2015. doi: 10.1039/C5CP00333D.

[92] Wei Hu, Lin Lin, and Chao Yang. Interpolative separable density fitting decomposition for accelerating hybrid density functional calculations with applications to defects in silicon. *J. Chem. Theory Comput.*, accepted, 2017. doi: 10.1021/acs.jctc.7b00807.

[93] Wei Hu, Lin Lin, and Chao Yang. Projected commutator diis method for accelerating hybrid functional electronic structure calculations. *J. Chem. Theory Comput.*, accepted, 2017. doi: 10.1021/acs.jctc.7b00892.

[94] Kejun Huang, Xiao Fu, and Nikolaos D. Sidiropoulos. Anchor-free correlated topic modeling: Identifiability and algorithm. In *NIPS*, 2016.

[95] B. Huffaker, M. Fomenkov, and k. claffy. Internet Topology Data Comparison.

Technical report, Cooperative Association for Internet Data Analysis (CAIDA), May 2012.

[96] Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 19(2):433–450, 1990.

[97] Dunham Jackson. *Über die Genauigkeit der Annäherung stetiger Funktionen durch ganze rationale Funktionen gegebenen Grades und trigonometrische Summen gegebener Ordnung*. Dieterich'schen Universität Buchdruckerei, 1911.

[98] John David Jackson. *Mathematics for Quantum Mechanics: An Introductory Survey of Operators, Eigenvalues, and Linear Vector Spaces*. Dover Publications, 2006.

[99] M. Jordan, Z. Ghahramani, T. Jaakola, and L. Saul. Introduction to variational methods for graphical models. *Machine Learning*, pages 183–233, 1999.

[100] Mark Kac. Can one hear the shape of a drum? *The American Mathematical Monthly*, 73(4):1–23, 1966.

[101] Leonid Vasilevich Kantorovich and Gennady S Rubinstein. On a space of completely additive functions. *Vestnik Leningrad. Univ*, 13(7):52–59, 1958.

[102] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. on Scientific Computing*, 20(1), 1998.

[103] Robert Keys. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(6):1153–1160, 1981.

[104] H. Koch, A. Sánchez de Merás, and T. B. Pedersen. Reduced scaling in electronic structure calculations using cholesky decompositions. *J. Chem. Phys.*, 118:9481–9484, 2003.

[105] Kurt Konolige, Motilal Agrawal, and Joan Sola. Large-scale visual odometry for rough terrain. In *Robotics Research*, pages 201–212. Springer, 2010.

[106] Alex Kulesza, Ben Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

[107] Alex Kulesza, N Raj Rao, and Satinder Singh. Low-rank spectral learning. In *Artificial Intelligence and Statistics*, pages 522–530, 2014.

[108] Thomas K Landauer, Peter W Foltz, and Darrell Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.

[109] Q. Le, T. Sarlos, and A. Smola. Fastfood-computing Hilbert space expansions in loglinear time. In *Proceedings of the 30th International Conference on Machine Learning*, pages 244–252, 2013.

[110] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788, 1999.

[111] Moontae Lee, David Bindel, and David Mimno. Robust spectral inference for joint stochastic matrix factorization. In *NIPS*, 2015.

[112] Moontae Lee, David Bindel, and David Mimno. From correlation to hierarchy: Practical topic modeling via spectral inference. In *12th INFORMS Workshop on Data Mining and Decision Analytics*, 2017.

[113] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

[114] Bruno Lévy. Laplace-Beltrami eigenfunctions towards an algorithm that "understands" geometry. In *Shape Modeling and Applications, 2006. SMI 2006. IEEE International Conference on*, pages 13–13. IEEE, 2006.

[115] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *NIPS*, 2014.

[116] Dawen Liang, Jaan Altosaar, Laurent Charlin, and David M Blei. Factorization meets the item embedding: Regularizing matrix factorization with item cooccurrence. In *Proceedings of the 10th ACM conference on recommender systems*, pages 59–66. ACM, 2016.

[117] L. Lin, J. Lu, L. Ying, and W. E. Adaptive local basis set for kohn-sham density functional theory in a discontinuous galerkin framework i: Total energy calculation. *J. Comput. Phys.*, 231(4):2140–2154, 2012. doi: 10.1016/j.jcp.2011.11.032.

[118] L. Lin, Z. Xu, and L. Ying. Adaptively compressed polarizability operator for accelerating large scale ab initio phonon calculations. *Multiscale Model. Simul.*, 15:29–55, 2017.

[119] Lin Lin. Adaptively compressed exchange operator. *J. Chem. Theory Comput.*, 12(5):2242–2249, 2016. doi: 10.1021/acs.jctc.6b00092.

[120] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[121] J. Lu and K. Thicke. Cubic scaling algorithms for RPA correlation using interpolative separable density fitting. *J. Comput. Phys.*, 351:187 – 202, 2017.

[122] J. Lu and L. Ying. Compression of the electron repulsion integral tensor in tensor hypercontraction format with cubic scaling cost. *J. Comput. Phys.*, 302:329–335, 2015. doi: 10.1016/j.jcp.2015.09.014.

[123] Ives Macedo, Joao Paulo Gois, and Luiz Velho. Hermite radial basis functions implicits. *Computer Graphics Forum*, 30(1):27–42, 2011.

[124] D MacKay and MN Gibbs. Efficient implementation of gaussian processes. *Neural Computation*, 1997.

[125] David JC MacKay. *Bayesian methods for adaptive models*. PhD thesis, California Institute of Technology, 1992.

[126] David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[127] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of the Fifth Berkeley Symp. On Math. Stat. and Prob.*, volume 1, pages 281–297, 1967.

[128] Jan R Magnus and Heinz Neudecker. Matrix differential calculus with applications in statistics and econometrics. *Wiley series in probability and mathematical statistics*, 1988.

[129] S. Manzer, P. R. Horn, N. Mardirossian, and M. Head-Gordon. Fast, accurate evaluation of exact exchange: The occ-RI-K algorithm. *J. Chem. Phys.*, 143(2): 024113, 2015.

[130] R. Martin. *Electronic Structure – Basic Theory and Practical Methods*. Cambridge Univ. Pr., West Nyack, NY, 2004.

[131] MATLAB Mapping Toolbox. MATLAB mapping toolbox, 2017. The MathWorks, Natick, MA, USA.

[132] H. P. McKean. Selberg's trace formula as applied to a compact riemann surface. *Communications on Pure and Applied Mathematics*, 25(3):225–246, 1972.

[133] Frank McSherry. Spectral partitioning of random graphs. In *focs*, page 529. IEEE, 2001.

[134] NN Medvedev. The algorithm for three-dimensional voronoi polyhedra. *Journal of computational physics*, 67(1):223–229, 1986.

[135] Erik H. W. Meijering, Karel J. Zuiderveld, and Max A. Viergever. Image reconstruction by convolution with symmetrical piecewise *n*th-order polynomial kernels. *IEEE Transactions on Image Processing*, 8(2):192–201, 1999.

[136] M. Mihail. Conductance and convergence of markov chains-a combinatorial treatment of expanders. In *30th Annual Symposium on Foundations of Computer Science*. IEEE, 1989. doi: 10.1109/sfcs.1989.63529. URL `https://doi.org/10.1109/sfcs.1989.63529`.

[137] Bojan Mohar. Isoperimetric numbers of graphs. *Journal of Combinatorial Theory, Series B*, 47(3):274–291, 1989.

[138] Cleve Moler and Charles Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM review*, 45(1):3–49, 2003.

[139] Ravi Montenegro, Prasad Tetali, et al. Mathematical aspects of mixing times in markov chains. *Foundations and Trends® in Theoretical Computer Science*, 1 (3):237–354, 2006.

[140] R. B. Murphy, M. D. Beachy, R. A. Friesner, and M. N. Ringnalda. Pseudospectral localized møller–plesset methods: Theory and calculation of conformational energies. *J. Chem. Phys.*, 103:1481, 1995.

[141] Radford M Neal. *Probabilistic inference using Markov chain Monte Carlo methods*, 1993.

[142] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

[143] Frank Neese, Frank Wennmohs, Andreas Hansen, and Ute Becker. Efficient, approximate and parallel hartree–fock and hybrid dft calculations. a "chain-of-spheres" algorithm for the hartree–fock exchange. *Chem. Phys.*, 356:98–109, 2009.

[144] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002.

[145] S. Nosé. A unified formulation of the constant temperature molecular dynamics methods. *J. Chem. Phys.*, 81:511, 1984. doi: 10.1063/1.447334.

[146] Robert L Ogniewicz and Olaf Kübler. Hierarchic voronoi skeletons. *Pattern recognition*, 28(3):343–359, 1995.

[147] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[148] B. N. Parlett. The software scene in the extraction of eigenvalues from sparse matrices. *SIAM Journal on Scientific and Statistical Computing*, 5(3):590–604, sep 1984. doi: 10.1137/0905042. URL `https://doi.org/10.1137/0905042`.

[149] R. M. Parrish, E. G. Hohenstein, T. J. Martínez, and C. D. Sherrill. Tensor hypercontraction. II. Least-squares renormalization. *J. Chem. Phys.*, 137:224106, 2012.

[150] R. M. Parrish, E. G. Hohenstein, T. J. Martínez, and C. D. Sherrill. Discrete

variable representation in electronic structure theory: Quadrature grids for least-squares tensor hypercontraction. *J. Chem. Phys.*, 138:194107, 2013.

[151] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global vectors for word representation. In *EMNLP*, 2014.

[152] Alex Pothen, Horst D. Simon, and Kan-Pu Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM J. Matrix Anal. Appl.*, 11(3):430–452, 1990.

[153] Jonathan K Pritchard, Matthew Stephens, and Peter Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155(2):945–959, 2000.

[154] Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.

[155] Joaquin Quinonero-Candela, Carl Edward Rasmussen, and Christopher KI Williams. Approximation methods for Gaussian process regression. *Large-scale kernel machines*, pages 203–223, 2007.

[156] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for Machine Learning*. The MIT Press, 2006.

[157] Carl Edward Rasmussen and Zoubin Ghahramani. Occam's razor. In *Neural Information Processing Systems (NIPS)*, 2001.

[158] Carl Edward Rasmussen and Hannes Nickisch. Gaussian processes for machine learning (GPML) toolbox. *Journal of Machine Learning Research (JMLR)*, 11: 3011–3015, Nov 2010.

[159] X. Ren, P. Rinke, V. Blum, J. Wieferink, A. Tkatchenko, A. Sanfilippo, K. Reuter, and M. Scheffler. Resolution-of-identity approach to Hartree-Fock, hybrid den-

sity functionals, RPA, MP2 and GW with numeric atom-centered orbital basis functions. *New J. Phys.*, 14:053020, 2012.

[160] G. Reynolds, T. J. Martinez, and E. A. Carter. Local weak pairs spectral and pseudospectral singles and doubles configuration interaction. *J. Chem. Phys.*, 105:6455, 1996.

[161] Christian Robert and George Casella. *Monte Carlo statistical methods*. Springer Science & Business Media, 2013.

[162] Havard Rue and Leonhard Held. *Gaussian Markov random fields: theory and applications*. CRC Press, 2005.

[163] Yousef Saad. *Numerical methods for large eigenvalue problems*. Manchester University Press, 1992.

[164] Robert Schaback and Holger Wendland. Kernel techniques: from machine learning to meshless methods. *Acta Numerica*, 15:543–639, 2006.

[165] Comandur Seshadhri, Tamara G Kolda, and Ali Pinar. Community structure and scale-free collections of erdős-rényi graphs. *Physical Review E*, 85(5):056109, 2012.

[166] Bernhard W Silverman. Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–52, 1985.

[167] Alistair Sinclair and Mark Jerrum. Approximate counting, uniform generation and rapidly mixing markov chains. *Information and Computation*, 82(1):93–133, jul 1989. doi: 10.1016/0890-5401(89)90067-9. URL `https://doi.org/10.1016/0890-5401(89)90067-9`.

[168] Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in neural information processing systems (NIPS)*, volume 18, page 1257. MIT Press, 2006.

[169] Michael L Stein, Jie Chen, Mihai Anitescu, et al. Stochastic approximation of score functions for gaussian processes. *The Annals of Applied Statistics*, 7(2): 1162–1191, 2013.

[170] S. Surjanovic and D. Bingham. Virtual library of simulation experiments: Test functions and datasets. http://www.sfu.ca/ ssurjano, 2018.

[171] A. Szabo and N.S. Ostlund. *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*. McGraw-Hill, New York, 1989.

[172] A. Tkatchenko and M. Scheffler. Accurate molecular van der waals interactions from ground-state electron density and free-atom reference data. *Phys. Rev. Lett.*, 102:073005, 2009.

[173] Lloyd N Trefethen. *Approximation theory and approximation practice*, volume 128. Siam, 2013.

[174] Luca Trevisan. Max cut and the smallest eigenvalue. *SIAM Journal on Computing*, 41(6):1769–1786, 2012.

[175] Shashanka Ubaru, Jie Chen, and Yousef Saad. Fast estimation of tr($F(A)$) via stochastic Lanczos quadrature.

[176] Madeleine Udell and Alex Townsend. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science*, 1(1):144–160, 2019.

[177] M. Wainwright and M. Jordan. Graphical models, exponential families, and vari-

ational inference. *Foundations and Trends in Machine Learning*, pages 1–305, 2008.

[178] Ziyu Wang, Masrour Zoghi, Frank Hutter, David Matheson, Nando De Freitas, et al. Bayesian optimization in high dimensions via random embeddings. In *Proceedings of the International Joint Conferences on Artificial Intelligence*, pages 1778–1784, 2013.

[179] Andrew J. Wathen and Shengxin Zhu. On spectral distribution of kernel matrices related to radial basis functions. *Numer. Algor.*, 70:709–726, 2015.

[180] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world' networks. *nature*, 393(6684):440, 1998.

[181] F. Weigend. A fully direct RI-HF algorithm: Implementation, optimised auxiliary basis sets, demonstration of accuracy and efficiency. *Phys. Chem. Chem. Phys.*, 4:4285–4291, 2002.

[182] Alexander Weiße, Gerhard Wellein, Andreas Alvermann, and Holger Fehske. The kernel polynomial method. *Reviews of modern physics*, 78(1), 2006.

[183] Holger Wendland. *Scattered data approximation*, volume 17. Cambridge university press, 2004.

[184] Hermann Weyl. Über die asymptotische verteilung der eigenwerte. *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse*, 1911:110–117, 1911.

[185] Hermann Weyl. Das asymptotische verteilungsgesetz der eigenwerte linearer partieller differentialgleichungen (mit einer anwendung auf die theorie der hohlraumstrahlung). *Mathematische Annalen*, 71(4):441–479, 1912.

[186] Eugene P. Wigner. On the distribution of the roots of certain symmetric matrices. *Annals of Mathematics*, pages 325–327, 1958.

[187] Andrew G Wilson, Zhiting Hu, Ruslan R Salakhutdinov, and Eric P Xing. Stochastic variational deep kernel learning. In *Advances in Neural Information Processing Systems*, pages 2586–2594, 2016.

[188] Andrew Gordon Wilson. *Covariance kernels for fast automatic pattern discovery and extrapolation with Gaussian processes*. PhD thesis, University of Cambridge, 2014.

[189] Andrew Gordon Wilson and Hannes Nickisch. Kernel interpolation for scalable structured Gaussian processes (KISS-GP). *International Conference on Machine Learning (ICML)*, 2015.

[190] Andrew Gordon Wilson, Elad Gilboa, Nehorai Arye, and John P Cunningham. Fast kernel learning for multidimensional pattern extrapolation. In *Advances in Neural Information Processing Systems*, pages 3626–3634, 2014.

[191] Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 370–378, 2016.

[192] Jian Wu, Matthias Poloczek, Andrew G Wilson, and Peter Frazier. Bayesian optimization with gradients. pages 5273–5284, 2017.

[193] G. Zhang, L. Lin, W. Hu, C. Yang, and J. E. Pask. Adaptive local basis set for Kohn–Sham density functional theory in a discontinuous Galerkin framework ii: Force, vibration, and molecular dynamics calculations. *J. Comput. Phys.*, 335: 426–443, 2017.